CICS® Transaction Server for OS/390®

CICS Performance Guide

Release 3

SC33-1699-33

CICS® Transaction Server for OS/390®

CICS Performance Guide

Release 3

SC33-1699-33

Note!
Before using this information and the product it supports, be sure to read the general information under "Notices" on page xiii.

Fourth edition (November 2000)

This edition applies to Release 3 of CICS Transaction Server for OS/390, program number 5655-147, and to all subsequent versions, releases, and modifications until otherwise indicated in new editions. Make sure you are using the correct edition for the level of the product.

This edition replaces and makes obsolete the previous edition, SC33-1699-33. Changes since that edition are indicated by a # sign to the left of a change. Any vertical lines in the left margin indicate a change made between Version 1 Release 2 and Version 1 Release 3 of CICS Transaction Server for OS/390.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address given below.

At the back of this publication is a page entitled "Sending your comments to IBM". If you want to make comments, but the methods described are not available to you, please address them to:

IBM United Kingdom Laboratories, Information Development,
Mail Point 95, Hursley Park, Winchester, Hampshire, England, SO21 2JN.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply in the United Kingdom or any other country where such provisions are inconsistent with local law:
INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore this statement may not apply to you.

This publication could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM United Kingdom Laboratories, MP151, Hursley Park, Winchester, Hampshire, England, SO21 2JN. Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Programming License Agreement, or any equivalent agreement between us.

Programming Interface Information

This book is intended to help you to:

- Establish performance objectives and monitor them
- Identify performance constraints, and make adjustments to the operational CICS system and its application programs.

This book also documents Product-sensitive Programming Interface and Associated Guidance Information and Diagnosis, Modification or Tuning Information provided by CICS.

Product-sensitive programming interfaces allow the customer installation to perform tasks such as diagnosing, modifying, monitoring, repairing, tailoring, or tuning of CICS. Use of such interfaces creates dependencies on the detailed design or implementation of the IBM software product. Product-sensitive programming interfaces should be used only for these specialized purposes. Because of their dependencies on detailed design and implementation, it is to be expected that programs written to such interfaces may need to be changed in order to run with new product releases or versions, or as a result of service.

Product-sensitive Programming Interface and Associated Guidance Information is identified where it occurs, either by an introductory statement to a chapter or section or by the following marking:

Product-sensitive programming interface

End of Product-sensitive programming interface

Diagnosis, Modification or Tuning Information is provided to help you tune your CICS system.

Attention: Do not use this Diagnosis, Modification or Tuning Information as a programming interface.

Diagnosis, Modification or Tuning Information is identified where it occurs, either by an introductory statement to a chapter or section or by the following marking:

Diagnosis, Modification or Tuning Information

End of Diagnosis, Modification or Tuning Information

Trademarks

The following terms are trademarks of International Business Machines Corporation in the United States, or other countries, or both:

ACF/VTAM	DFSMS/MVS	NetView
CICS	GDDM	OS/2
CICS/ESA	Hiperspace	OS/390
CICS/MVS	IBM	RACF
CICSPlex SM	IMS/ESA	RMF
DATABASE 2	MVS/DFP	System/390
DB2	MVS/ESA	VTAM

Other company, product, and service names may be trademarks or service marks of others.

Preface

What this book is about

This book is intended to help you to:

- Establish performance objectives and monitor them
- Identify performance constraints, and make adjustments to the operational CICS system and its application programs.

This book does not discuss the performance aspects of the CICS Transaction Server for OS/390 Release 3 Front End Programming Interface. For more information about the Front End Programming Interface, See the *CICS Front End Programming Interface User's Guide*. This book does not contain Front End Programming Interface dump statistics.

Who this book is for

This book is for a person who is involved in:

- System design
- Monitoring and tuning CICS® performance.

What you need to know to understand this book

You need to have a good understanding of how CICS works. This assumes familiarity with many of the books in the CICS Transaction Server for OS/390 Release 3 library, together with adequate practical experience of installing and maintaining a CICS system.

How to use this book

If you want to establish performance objectives, monitor the performance of a CICS system, and occasionally make adjustments to the system to keep it within objectives, you should read through this book in its entirety.

If you have a performance problem and want to correct it, read Parts 3 and 4. You may need to refer to various sections in Part 2.

Notes on terminology

The following abbreviations are used throughout this book:

- "CICS" refers to the CICS element in the CICS Transaction Server for OS/390®
- "MVS" refers to the operating system, which can be either an element of OS/390, or MVS/Enterprise System Architecture System Product (MVS/ESA SP).
- "VTAM®" refers to ACF/VTAM.
- "DL/I" refers to the database component of IMS/ESA.

Bibliography

CICS Transaction Server for OS/390

CICS Transaction Server for OS/390: Planning for Installation	GC33-1789
CICS Transaction Server for OS/390 Release Guide	GC34-5352
CICS Transaction Server for OS/390 Migration Guide	GC34-5353
CICS Transaction Server for OS/390 Installation Guide	GC33-1681
CICS Transaction Server for OS/390 Program Directory	GI10-2506
CICS Transaction Server for OS/390 Licensed Program Specification	GC33-1707

CICS books for CICS Transaction Server for OS/390

General	
CICS Master Index	SC33-1704
CICS User's Handbook	SX33-6104
CICS Transaction Server for OS/390 Glossary (softcopy only)	GC33-1705
Administration	
CICS System Definition Guide	SC33-1682
CICS Customization Guide	SC33-1683
CICS Resource Definition Guide	SC33-1684
CICS Operations and Utilities Guide	SC33-1685
CICS Supplied Transactions	SC33-1686
Programming	
CICS Application Programming Guide	SC33-1687
CICS Application Programming Reference	SC33-1688
CICS System Programming Reference	SC33-1689
CICS Front End Programming Interface User's Guide	SC33-1692
CICS C++ OO Class Libraries	SC34-5455
CICS Distributed Transaction Programming Guide	SC33-1691
CICS Business Transaction Services	SC34-5268
Diagnosis	
CICS Problem Determination Guide	GC33-1693
CICS Messages and Codes	GC33-1694
CICS Diagnosis Reference	LY33-6088
CICS Data Areas	LY33-6089
CICS Trace Entries	SC34-5446
CICS Supplementary Data Areas	LY33-6090
Communication	
CICS Intercommunication Guide	SC33-1695
CICS Family: Interproduct Communication	SC33-0824
CICS Family: Communicating from CICS on System/390	SC33-1697
CICS External Interfaces Guide	SC33-1944
CICS Internet Guide	SC34-5445
Special topics	
CICS Recovery and Restart Guide	SC33-1698
CICS Performance Guide	SC33-1699
CICS IMS Database Control Guide	SC33-1700
CICS RACF Security Guide	SC33-1701
CICS Shared Data Tables Guide	SC33-1702
CICS Transaction Affinities Utility Guide	SC33-1777

CICS DB2 Guide	SC33-1939

CICSPlex SM books for CICS Transaction Server for OS/390

General	
CICSPlex SM Master Index	SC33-1812
CICSPlex SM Concepts and Planning	GC33-0786
CICSPlex SM User Interface Guide	SC33-0788
CICSPlex SM Web User Interface Guide	SC34-5403
CICSPlex SM View Commands Reference Summary	SX33-6099
Administration and Management	
CICSPlex SM Administration	SC34-5401
CICSPlex SM Operations Views Reference	SC33-0789
CICSPlex SM Monitor Views Reference	SC34-5402
CICSPlex SM Managing Workloads	SC33-1807
CICSPlex SM Managing Resource Usage	SC33-1808
CICSPlex SM Managing Business Applications	SC33-1809
Programming	
CICSPlex SM Application Programming Guide	SC34-5457
CICSPlex SM Application Programming Reference	SC34-5458
Diagnosis	
CICSPlex SM Resource Tables Reference	SC33-1220
CICSPlex SM Messages and Codes	GC33-0790
CICSPlex SM Problem Determination	GC33-0791

Other CICS books

CICS Application Programming Primer (VS COBOL II)	SC33-0674
CICS Application Migration Aid Guide	SC33-0768
CICS Family: API Structure	SC33-1007
CICS Family: Client/Server Programming	SC33-1435
CICS Family: General Information	GC33-0155
CICS 4.1 Sample Applications Guide	SC33-1173
CICS/ESA 3.3 XRF Guide	SC33-0661

If you have any questions about the CICS Transaction Server for OS/390 library, see *CICS Transaction Server for OS/390: Planning for Installation* which discusses both hardcopy and softcopy books and the ways that the books can be ordered.

Books from related libraries

ACF/VTAM

ACF/VTAM Installation and Migration Guide, GC31-6547-01
ACF/VTAM Network Implementation Guide, SC31-6548

CICSPlex System Manager for MVS/ESA

IBM CICSPlex System Manager for MVS/ESA Setup and Administration - Volume 1, SC33-0784-01
IBM CICSPlex System Manager for MVS/ESA Setup and Administration - Volume 2, SC33-0784-02

DATABASE 2

DB2 for OS/390 Administration Guide, SC26-8957

DATABASE 2 Performance Monitor (DB2PM)

DB2 PM Batch User's Guide, SH12-6164
DB2 PM Command Reference, SH12-6167
DB2 PM Online Monitor User's Guide, SH12-6165
DB2 PM Report Reference, SH12-6163
DB2 for OS/390 Capacity Planning, SG24-2244
DB2 PM Usage Guide Update, SG24-2584

DFSMS/MVS

DFSMS/MVS NaviQuest User's Guide, SC26-7194
DFSMS/MVS DFSMSdfp Storage Administration Reference, SC26-4920

IMS/ESA

IMS/ESA Version 5 Admin Guide: DB, SC26-8012
IMS/ESA Version 5 Admin Guide: System, SC26-8013
IMS/ESA Version 5 Performance Analyzer's User's Guide, SC26-9088
IMS/ESA Version 6 Admin Guide: DB, SC26-8725
IMS/ESA Version 6 Admin Guide: System, SC26-8720
IMS Performance Analyzer User's Guide SC26-9088

MVS

OS/390 MVS Initialization and Tuning Guide, SC28-1751
OS/390 MVS Initialization and Tuning Reference, SC28-1752
OS/390 MVS JCL Reference, GC28-1757
OS/390 MVS System Management Facilities (SMF), GC28-1783
OS/390 MVS Planning: Global Resource Serialization, GC28-1759
OS/390 MVS Planning: Workload Management, GC28-1761
OS/390 MVS Setting Up a Sysplex, GC28-1779

OS/390 RMF

OS/390 RMF User's Guide, GC28-1949-01
OS/390 Performance Management Guide, SC28-1951-00
OS/390 RMF Report Analysis, SC28-1950-01
OS/390 RMF Programmers Guide, SC28-1952-01

Tivoli Performance Reporter for OS/390

Tivoli Performance Reporter for OS/390: Administration Guide, SH19-6816
Tivoli Performance Reporter for OS/390: CICS Performance Feature Guide and Reference, SH19-6820
SLR to Tivoli Performance Reporter for OS/390: Migration Cookbook, SG24-5128

NetView Performance Monitor (NPM)

NPM Reports and Record Formats, SH19-6965-01
NPM User's Guide, SH19-6962-01

Tuning tools

Generalized Trace Facility Performance Analysis (GTFPARS) Program Description/Operations Manual, SB21-2143
Network Performance Analysis and Reporting System Program Description/Operations, SB21-2488

Network Program Products Planning, SC30-3351

Others

CICS Workload Management Using CICSPlex SM and the MVS/ESA Workload Manager, GG24-4286
System/390 MVS Parallel Sysplex Performance, GG24-4356
System/390 MVS/ESA Version 5 Workload Manager Performance Studies, SG24-4352
IBM 3704 and 3705 Control Program Generation and Utilities Guide, GC30-3008
IMSASAP II Description/Operations, SB21-1793
Screen Definition Facility II Primer for CICS/BMS Programs, SH19-6118
Systems Network Architecture Management Services Reference,SC30-3346
Teleprocessing Network Simulator General Information, GH20-2487

Determining if a publication is current

IBM regularly updates its publications with new and changed information. When first published, both hardcopy and BookManager softcopy versions of a publication are usually in step. However, due to the time required to print and distribute hardcopy books, the BookManager version is more likely to have had last-minute changes made to it before publication.

Subsequent updates will probably be available in softcopy before they are available in hardcopy. This means that at any time from the availability of a release, softcopy versions should be regarded as the most up-to-date.

For CICS Transaction Server books, these softcopy updates appear regularly on the *Transaction Processing and Data Collection Kit* CD-ROM, SK2T-0730-xx. Each reissue of the collection kit is indicated by an updated order number suffix (the -xx part). For example, collection kit SK2T-0730-06 is more up-to-date than SK2T-0730-05. The collection kit is also clearly dated on the cover.

Updates to the softcopy are clearly marked by revision codes (usually a "#" character) to the left of the changes.

Summary of changes

Changes since CICS Transaction Server for OS/390 Release 2 are indicated by vertical lines to the left of the text.

Changes for CICS Transaction Server for OS/390 Release 3

The chapter on Service Level Reporter(SLR) has been removed.

"Chapter 7. Tivoli Performance Reporter for OS/390" on page 109 replaces the chapter on Performance Reporter for MVS..

Performance considerations resulting from enhancements to CICS Web support and the introduction of Secure Sockets Layer for Web security, are discussed in "Chapter 18. CICS Web support" on page 217.

The performance implications of using Coupling Facilities Data Tables, including information about contention model and locking model, are discussed in "Chapter 19. VSAM and file control" on page 221.

A chapter has been added, "Chapter 20. Java program objects" on page 251, to introduce performance considerations when using Java language support.

"Chapter 21. Java virtual machine (JVM) programs" on page 257 describes performance implications for programs run using the MVS Java Virtual Machine (JVM).

"Chapter 8. Managing Workloads" on page 119 has been revised to discuss more fully the implications and benefits of using the MVS workload manager, and to introduce the CICSPlex SM dynamic routing program used by the WLM.

Additional or changed statistics for the following have been documented:

- Dispatcher domain
- Enqueue domain
- Files
- ISC/IRC
- TCP/IP Services

Separate appendixes have been created to show the statistics obtained for the following:

- Coupling facility data tables server
- Named counter sequence number server

Changes have also been made to several reports in the sample statistics program, DFH0STAT.

Changes for CICS Transaction Server for OS/390 Release 2

- The CICS DB2 attachment facility supplied with CICS Transaction Server for OS/390 Release 2 provides resource definition online (RDO) support for DB2 resources as an alternative to resource definition table (RCT) definitions. CICS DB2 statistics, collected using standard CICS interfaces, are provided in Appendix A.
- "Chapter 22. Database management" on page 261 discusses relevant parameters of the CICS DB2 attachment facility.
- Information about tuning the performance of DASD-only log streams has been added to "Chapter 23. Logging and journaling" on page 269.
- A full description of User Domain statistics is provided.
- Additions have been made to performance data for groups DFHFILE, DFHPROG, DFHTASK, and DFHTEMP in "Chapter 6. The CICS monitoring facility" on page 61.
- "Appendix F. MVS and CICS virtual storage" on page 611 has an additional section, "Short-on-storage conditions caused by subpool storage fragmentation" on page 631.

Changes for CICS Transaction Server Release 1

- As part of the restructure of the temporary storage section, the TSMGSET system initialization parameter has been deleted.
- The XRF function has not changed for CICS Transaction for OS/390 Release 1, but the chapter, Tuning XRF, has been removed from this book. For information about tuning XRF see the CICS/ESA 4.1 edition of the *Performance Guide*.
- For VSAM RLS files, the IMBED cluster attribute has been withdrawn, and the REPLICATE cluster is no longer recommended. You can achieve the effects of Imbed and Replication by using caching controllers.
- The enterprise performance data manager introduced in CICS Transaction Server for OS/390 Release 1 has been renamed Performance Reporter for MVS. See *Chapter 7.*
- The role of the system initialization parameters, DSHIPINT and DSHIPIDL is discussed in "Chapter 25. MRO and ISC" on page 301.
- Information about automatic restart management (ARM), as a sysplex-wide restart mechanism is given in "Chapter 28. Improving CICS startup and normal shutdown time" on page 333.
- Journal control statistics have been replaced by Journalname statistics and Log Stream statistics. They represent the activity on journals within MVS log streams and SMF data sets. See "Journalname" on page 405, and "Log stream" on page 407.
- An Appendix has been added to explain the shared temporary storage server statistics that are produced when determining how much available storage can be allocated to the server. See *Appendix B. Shared temporary storage queue server statistics.*
- A temporary storage domain has been introduced, and a number of TSMAIN subpools are to be added to the list of CICS subpools in the ECDSA in "Appendix F. MVS and CICS virtual storage" on page 611.
- A different methodology has been used to produce the latest data presented in "Appendix G. Performance data" on page 637.

Changes for the CICS/ESA 4.1 edition

Changes for the CICS/ESA Version 4 Release 1 edition include the following:

- Additional or changed statistics in the following areas have been documented:
  - Autoinstalled statistics
  - DBCTL statistics
  - Dispatcher statistics
  - DL/I statistics
  - FEPI pool statistics
  - FEPI connection statistics
  - FEPI target statistics
  - File control statistics
  - ISC/IRC system and mode entry statistics
  - Journal control statistics
  - Loader statistics
  - LSR pool statistics
  - Program autoinstalled statistics
  - Storage manager statistics
  - Suspending mirrors and MROLM
  - Terminal control statistics
  - Terminal autoinstalled statistics
  - Transaction statistics
  - Transaction class statistics
  - Transaction manager statistics
  - Transient data statistics
  - VTAM statistics.
- The domain manager statistics have been removed from this release.
- The description of the data produced by the CICS monitoring facility has been transferred from the *Customization Guide* and is included in "Interpreting CICS monitoring" on page 68.
- "Chapter 9. Understanding RMF workload manager data" on page 133 has been added to explain CICS-related data in an RMF workload activity report.
- "Use of LLA (MVS library lookaside)" on page 192 includes a section on persistent sessions delay interval (PSINT).
- "Intersystems session queue management" on page 303 has been added to "Chapter 25. MRO and ISC" on page 301.
- A new appendix has been added giving details of the sample statistics program (DFH0STAT). See "Appendix E. The sample statistics program, DFH0STAT" on page 513.
- The storage chapter has been modified, and a new section about kernel storage has been added in "CICS kernel storage" on page 634.

Part 1. Setting performance objectives

This book describes how CICS performance might be improved. It also provides reference information to help you achieve such improvement.

Good performance is the achievement of agreed service levels. This means that system availability and response times meet user's expectations using resources available within the budget.

The performance of a CICS system should be considered:

- When you plan to install a new system
- When you want to review an existing system
- When you contemplate major changes to a system.

There are several basic steps in tuning a system, some of which may be just iterative until performance is acceptable. These are:

1. Agree what good performance is.
2. Set up performance objectives (described in Chapter 1. Establishing performance objectives).
3. Decide on measurement criteria (described in Chapter 3. Performance monitoring and review).
4. Measure the performance of the production system.
5. Adjust the system as necessary.
6. Continue to monitor the performance of the system and anticipate future constraints (see "Monitoring for the future" on page 13).

Parts 1 and 2 of this book describe how to monitor and assess performance.

Parts 3 and 4 suggest ways to improve performance.

This part contains the following chapters:

Recommendations given in this book, based on current knowledge of CICS, are general in nature, and cannot be guaranteed to improve the performance of any particular system.

Chapter 1. Establishing performance objectives

The process of establishing performance objectives is described in this chapter in the following sections:

Performance objectives often consist of a list of transactions and expected timings for each. Ideally, through them, good performance can be easily recognized and you know when to stop further tuning. They must, therefore, be:

- Practically measurable
- Based on a realistic workload
- Within the budget.

Such objectives may be defined in terms such as:

- Desired or acceptable response times, for example, within which 90% of all responses occur
- Average or peak number of transactions through the system
- System availability, including mean time to failure, and downtime after a failure.

After you have defined the workload and estimated the resources required, you must reconcile the desired response with what you consider attainable. These objectives must then be agreed and regularly reviewed with users.

Establishing performance objectives is an iterative process involving the activities described in the rest of this chapter.

Defining some terms

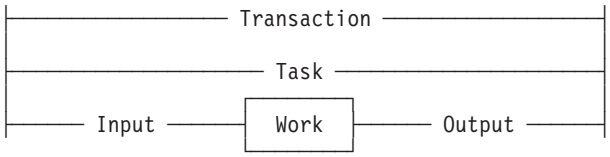
For performance measurements we need to be very specific about what we are measuring. Therefore, it is necessary to define a few terms.

The word *user* here means the terminal operator. A user, so defined, sees CICS performance as the *response time*, that is, the time between the last input action (for example, a keystroke) and the expected response (for example, a message on the screen). Several such responses might be required to complete a user *function*, and the amount of work that a user perceives as a function can vary enormously. So, the number of functions per period of time is not a good measure of performance, unless, of course, there exists an agreed set of benchmark functions.

A more specific unit of measure is therefore needed. The words *transaction* and *task* are used to describe units of work within CICS. Even these can lead to ambiguities, because it would be possible to define transactions and tasks of varying size. However, within a particular system, a series of transactions can be well defined and understood so that it becomes possible to talk about relative performance in terms of transactions per second (or minute, or hour).

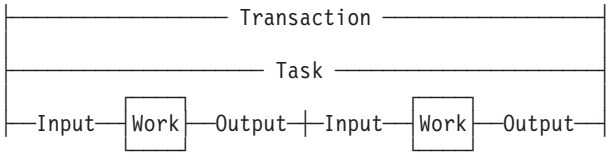
In this context there are three modes of CICS operation.

Nonconversational



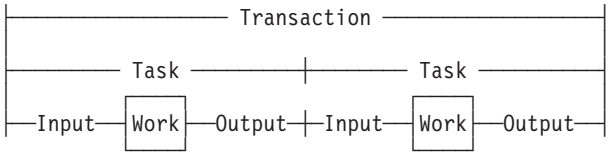
Nonconversational mode is of the nature of one question, one answer; resources are allocated, used, and released immediately on completion of the task. In this mode the words transaction and task are more or less synonymous.

Conversational



Conversational mode is potentially wasteful in a system that does not have abundant resources. There are further questions and answers during which resources are not released. Resources are, therefore, tied up unnecessarily waiting for users to respond, and performance may suffer accordingly. Transaction and task are, once again, more or less synonymous.

Pseudoconversational



Pseudoconversational mode allows for slow response from the user. Transactions are broken up into more than one task, yet the user need not know this. The resources in demand are released at the end of each task, giving a potential for improved performance.

The input/output surrounding a task may be known as the *dialog*.

Defining performance objectives and priorities

Performance objectives and priorities depend on user's expectations. From the point of view of CICS, these objectives state response times to be seen by the terminal user, and the total throughput per day, hour, or minute.

The first step in defining performance objectives is to specify what is required of the system. In doing this, you must consider the available hardware and software resources so that reasonable performance objectives can be agreed. Alternatively you should ascertain what additional resource is necessary to attain users' expectations, and what that resource would cost. This cost might be important in negotiations with users to reach an acceptable compromise between response time and required resource.

An agreement on acceptable performance criteria between the data processing and user groups in an organization is often formalized and called a *service level agreement*.

Common examples in these agreements are, on a network with remote terminals, that 90% of all response times sampled are under six seconds in the prime shift, or that the average response time does not exceed 12 seconds even during peak periods. (These response times could be substantially lower in a network consisting only of local terminals.)

You should consider whether to define your criteria in terms of the average, the 90th percentile, or even the worst-case response time. Your choice may depend on the audit controls of your installation and the nature of the transactions in question.

Analyzing the current workload

Break down the work to be done into transactions. Develop a profile for each transaction that includes:

- The *workload*, that is, the amount of work done by CICS to complete this transaction. In an ideal CICS system (with optimum resources), most transactions perform a single function with an identifiable workload.
- The *volume*, that is, the number of times this transaction is expected to be executed during a given period. For an active system, you can get this from the CICS statistics.

Later, transactions with common profiles can be merged, for convenience into *transaction categories*.

Establish the priority of each transaction category, and note the periods during which the priorities change.

Determine the resources required to do the work, that is:

- Physical resources managed by the operating system (real storage, DASD I/O, terminal I/O)
- Logical resources managed by the subsystem, such as control blocks and buffers.

To determine transaction resource demands, you can make sample measurements on a dedicated machine using the CICS monitoring facility. Use these results to suggest possible changes that could have the greatest effect if applied before system-wide contention arises. You can also compare your test results with those in the production environment.

See “Chapter 2. Gathering data for performance objectives” on page 7 for more detailed recommendations on this step.

Translating resource requirements into system objectives

You have to translate the information you have gathered into system-oriented objectives for each transaction category. Such objectives include statements about the transaction volumes to be supported (including any peak periods) and the response times to be achieved.

Any assumptions that you make about your installation must be used consistently in future monitoring. These assumptions include *computing-system factors* and *business factors*.

Computing-system factors include the following:

- *System response time*: this depends on the design and implementation of the code, and the power of the processor.
- *Network response time*: this can amount to seconds, while responses in the processor are likely to be in fractions of seconds. This means that a system can never deliver good responses through an overloaded network, however good the processor.
- *DASD response time*: this is generally responsible for most of the internal processing time required for a transaction. You must consider all I/O operations that affect a transaction.
- *Existing workload*: this may affect the performance of new transactions, and vice versa. In planning the capacity of the system, consider the total load on each major resource, not just the load for the new application.

  Response times can vary for a number of reasons, and the targets should, therefore, specify an acceptable degree of tolerance. Allow for transactions that are known to make heavy demands on the processor and database I/O.

  To reconcile expectations with performance, it may be necessary to change the expectations or to vary the mix or volume of transactions.

Business factors are concerned with work fluctuations. Allow for daily peaks (for example, after receipt of mail), weekly peaks (for example, Monday peak after weekend mail), and seasonal peaks as appropriate to the business. Also allow for the peaks of work after planned interruptions, such as preventive maintenance and public holidays.

Chapter 2. Gathering data for performance objectives

During the design, development, and test of a total system, information is gathered about the complexity of processing with particular emphasis on I/O activity. This information is used for establishing performance objectives.

The following phases of installation planning are discussed in this chapter:

Requirements definition phase

In this phase, careful estimates are your only input, as follows:

- Number of transactions for each user function
- Number of I/O operations per user function (DASD and terminals)
- Time required to key in user data (including user "thinking time")
- Line speeds (number of characters per second) for remote terminals
- Number of terminals and operators required to achieve the required rate of input
- Maximum rate of transactions per minute/hour/day/week
- Average and maximum workloads (that is, processing per transaction)
- Average and maximum volumes (that is, total number of transactions)
- Likely effects of performance objectives on operations and system programming.

External design phase

During the external design phase, you should:

1. Estimate the network, processor, and DASD loading based on the dialog between users and tasks (that is, the input to each transaction, and consequent output).
2. Revise your disk access estimates. After external design, only the logical data accesses are defined (for example, EXEC CICS READ).
3. Estimate coupling facility resources usage for the MVS system logger and resource files, or any cross-system coupling facility (XCF) activity.

Remember that, after the system has been brought into service, no amount of tuning can compensate for poor initial design.

Internal design phase

More detailed information is available to help:

- Refine your estimate of loading against the work required for each transaction dialog. Include screen control characters for field formatting.

- Refine disk access estimates against database design. After internal design, the physical data accesses can be defined at least for the application-oriented accesses.
- Add the accesses for CICS temporary storage (scratchpad) data, program library, and CICS transient data to the database disk accesses.
- Consider if additional loads could cause a significant constraint.
- Refine estimates on processor use.

Coding and testing phase

During the coding and testing phase, you should:

1. Refine the internal design estimates of disk and processing resources.
2. Refine the network loading estimates.
3. Run the monitoring tools and compare results with estimates. See "Chapter 4. An overview of performance-measurement tools" on page 21 for information on the CICS monitoring tools.

Post-development review

Review the performance of the complete system in detail. The main purposes are to:

- Validate performance against objectives
- Identify resources whose use requires regular monitoring
- Feed the observed figures back into future estimates.

To achieve this, you should:

1. Identify discrepancies from the estimated resource use
2. Identify the categories of transactions that have caused these discrepancies
3. Assign priorities to remedial actions
4. Identify resources that are consistently heavily used
5. Provide utilities for graphic representation of these resources
6. Project the loadings against the planned future system growth to ensure that adequate capacity is available
7. Update the design document with the observed performance figures
8. Modify the estimating procedures for future systems.

Information supplied by end users

Comments from users are a necessary part of the data for performance analysis and improvement. Reporting procedures must be established, and their use encouraged.

Log exceptional incidents. These incidents should include system, line, or transaction failure, and response times that are outside specified limits. In addition, you should log incidents that threaten performance (such as deadlocks, deadlock abends, stalls, indications of going short-on-storage (SOS) and maximum number of multiregion operation (MRO) sessions used) as well as situations such as recoveries, including recovery from DL/I deadlock abend and restart, which mean that additional system resources are being used.

The data logged should include the date and time, location, duration, cause (if known), and the action taken to resolve the problem.

Chapter 3. Performance monitoring and review

This chapter describes in the following sections some monitoring techniques; and how to use them.

Once set, as described in "Chapter 1. Establishing performance objectives" on page 3, performance objectives should be monitored using appropriate methods.

Deciding on monitoring activities and techniques

In this book, *monitoring* is specifically used to describe regular checking of the performance of a CICS production system, against objectives, by the collection and interpretation of data. Subsequently, *analysis* describes the techniques used to investigate the reasons for performance deterioration. *Tuning* may be used for any actions that result from this analysis.

Monitoring should be ongoing because it:

- Establishes transaction profiles (that is, workload and volumes) and statistical data for predicting system capacities
- Gives early warning through comparative data to avoid performance problems
- Measures and validates any tuning you may have done in response to an earlier performance problem.

A performance history database (see "Tivoli Performance Reporter for OS/390" on page 29 for an example) is a valuable source from which to answer questions on system performance, and to plan further tuning.

Monitoring may be described in terms of strategies, procedures, and tasks.

Strategies may include:

- Continuous or periodic summaries of the workload. You can track all transactions or selected representatives.
- Snapshots at normal or peak loads. Peak loads should be monitored for two reasons:
  1. Constraints and slow responses are more pronounced at peak volumes.
  2. The current peak load is a good indicator of the future average load.

Procedures, such as good documentation practices, should provide a management link between monitoring strategies and tasks. The following should be noted:

- The growth of transaction rates and changes in the use of applications
- Consequent extrapolation to show possible future trends
- The effects of nonperformance system problems such as application abends, frequent signon problems, and excessive retries.

Tasks (not to be confused with the task component of a CICS transaction) include:

- Running one or more of the tools described in "Chapter 4. An overview of performance-measurement tools" on page 21
- Collating the output
- Examining it for trends.

You should allocate responsibility for these tasks between operations personnel, programming personnel, and analysts. You must identify the resources that are to be regarded as critical, and set up a procedure to highlight any trends in the use of these resources.

Because the tools require resources, they may disturb the performance of a production system.

Give emphasis to peak periods of activity, for both the new application and the system as a whole. It may be necessary to run the tools more frequently at first to confirm that the expected peaks correspond with the actual ones.

It is not normally practical to keep all the detailed output. Arrange for summarized reports to be filed with the corresponding CICS statistics, and for the output from the tools to be held for an agreed period, with customary safeguards for its protection.

Conclusions on performance should not be based on one or two snapshots of system performance, but rather on data collected at different times over a prolonged period. Emphasis should be placed on peak loading. Because different tools use different measurement criteria, early measurements may give apparently discrepant results.

Your monitoring procedures should be planned ahead of time. These procedures should explain the tools to be used, the analysis techniques to be used, the operational extent of those activities, and how often they are to be performed.

Developing monitoring activities and techniques

When you are developing a master plan for monitoring and performance analysis, you should establish:

- A master schedule of monitoring activity. You should coordinate monitoring with operations procedures to allow for feedback of online events as well as instructions for daily or periodic data gathering.
- The tools to be used for monitoring. The tools used for data gathering should provide for dynamic monitoring, daily collection of statistics, and more detailed monitoring. (See "When to review?" on page 11.)
- The kinds of analysis to be performed. This must take into account any controls you have already established for managing the installation, for example, the use of the Performance Reporter, and so on. You should document what data is to be extracted from the monitoring output, identifying the source and usage of the

data. Although the formatted reports provided by the monitoring tools help to organize the volume of data, you may need to design worksheets to assist in data extraction and reduction.

- A list of the personnel who are to be included in any review of the findings. The results and conclusions from analyzing monitor data should be made known to the user liaison group and to system performance specialists.
- A strategy for implementing changes to the CICS system design resulting from tuning recommendations. This has to be incorporated into installation management procedures, and would include items such as standards for testing and the permitted frequency of changes to the production environment.

Planning the review process

Establish a schedule for monitoring procedures. This schedule should be as simple as possible. The activities done as part of the planning should include the following:

- Listing the CICS requests made by each type of task. This helps you decide which requests or which resources (the high-frequency or high-cost ones) need to be looked at in statistics and CICS monitoring facility reports.
- Drawing up checklists of review questions.
- Estimating resource usage and system loading for new applications. This is to enable you to set an initial basis from which to start comparisons.

When to review?

You should plan for the following broad levels of monitoring activity:

- Dynamic (online) monitoring.
- Daily monitoring.
- Periodic (weekly and monthly) monitoring.
- Keeping sample reports as historical data. You can also keep historical data in a database such as the Performance Reporter database.

Dynamic monitoring

Dynamic monitoring, is "on-the-spot" monitoring that you can, and should, carry out at all times. This type of monitoring generally includes the following:

- Observing the system's operation continuously to discover any serious short-term deviation from performance objectives.

  Use the CEMT transaction (CEMT INQ|SET MONITOR), together with end-user feedback. You can also use the Resource Measurement Facility (RMF) to collect information about processor, channel, coupling facility, and I/O device usage.
- Obtaining status information. Together with status information obtained by using the CEMT transaction, you can get status information on system processing during online execution. This information could include the queue levels, active regions, active terminals, and the number and type of conversational transactions. You could get this information with the aid of an automated program invoked by the master terminal operator. At prearranged times in the production cycle (such as before scheduling a message, at shutdown of part of the network, or at peak loading), the program could capture the transaction processing status and measurements of system resource levels.
- The System Management product, CICSPlex® SM, can accumulate information produced by the CICS monitoring facility to assist in dynamic monitoring activities. The data can then be immediately viewed online, giving instant

feedback on the performance of the transactions. To allow CICSPlex SM to collect CICS monitoring information, CICS monitoring must be active using CEMT SET MONITOR ON.

Daily monitoring

The overall objective here is to measure and record key system parameters daily. The daily monitoring data usually consists of counts of events and gross level timings. In some cases, the timings are averaged for the entire CICS system.

- Record both the daily average and the peak period (usually one hour) average of, for example, messages, tasks, processor usage, I/O events, and storage used. Compare these against your major performance objectives and look for adverse trends.
- List the CICS-provided statistics at the end of every CICS run. You should date and time-stamp the data that is provided, and file it for later review. For example, in an installation that has settled down, you might review daily data at the end of the week; generally, you can carry out reviews less frequently than collection, for any one type of monitoring data. If you know there is a problem, you might increase the frequency; for example, reviewing daily data immediately it becomes available.

  You should be familiar with all the facilities in CICS for providing statistics at times other than at shutdown. The main facilities, using the CEMT transaction, are invocation from a terminal (with or without reset of the counters) and automatic time-initiated requests.
- File an informal note of any incidents reported during the run. These may include a shutdown of CICS that causes a gap in the statistics, a complaint from your end users of poor response times, a terminal going out of service, or any other item of significance. This makes it useful when reconciling disparities in detailed performance figures that may be discovered later.
- Print the system console log for the period when CICS was active, and file a copy of the console log in case it becomes necessary to review the CICS system performance in the light of the concurrent batch activity.
- Run one of the performance analysis tools described in “Chapter 4. An overview of performance-measurement tools” on page 21 for at least part of the day if there is any variation in load from day to day. File the summaries of the reports produced by the tools you use.
- Transcribe onto a graph any items identified as being consistently heavily used in the post-development review phase (described in “Chapter 2. Gathering data for performance objectives” on page 7).
- Collect CICS statistics, monitoring data, and RMF™ data into the Performance Reporter database.

Weekly monitoring

Here, the objective is to periodically collect detailed statistics on the operation of your system for comparison with your system-oriented objectives and workload profiles.

- Run the CICS monitoring facility with performance class active, and process it. It may not be necessary to do this every day, but it is important to do it regularly and to keep the sorted summary output as well as the detailed reports.

  Whether you do this on the same day of the week depends on the nature of the system load. If there is an identifiable heavy day of the week, this is the one that you should monitor. (Bear in mind, however, that the use of the monitoring facility causes additional load, particularly with performance class active.)

If the load is apparently the same each day, run the CICS monitoring facility daily for a period sufficient to confirm this. If there really is little difference from day to day in the CICS load, check the concurrent batch loads in the same way from the logs. This helps you identify any obscure problems because of peak volumes or unusual transaction mixes on specific days of the week. The first few weeks' output from the CICS statistics also give guidance for this.

It may not be necessary to review the detailed monitor report output every time, but you should always keep this output in case the summary data is insufficient to answer questions raised by the statistics or by user comments. Label the CICS monitoring facility output tape (or a dump of the DASD data set) and keep it for an agreed period in case further investigations are required.

- Run RMF, because this shows I/O usage, channel usage, and so on. File the summary reports and archive the output tapes for some agreed period.
- Review the CICS statistics, and any incident reports.
- Review the graph of critical parameters. If any of the items is approaching a critical level, check the performance analysis and RMF outputs for more detail and follow any previously agreed procedures (for example, notify your management).
- Tabulate or produce a graph of values as a summary for future reference.
- Produce weekly Performance Reporter reports.

Monthly monitoring

- Run RMF.
- Review the RMF and performance analysis listings. If there is any indication of excessive resource usage, follow any previously agreed procedures (for example, notify your management), and do further monitoring.
- Date- and time-stamp the RMF output and keep it for use in case performance problems start to arise. You can also use the output in making estimates, when detailed knowledge of component usage may be important. These aids provide detailed data on the usage of resources within the system, including processor usage, use of DASD, and paging rates.
- Produce monthly Performance Reporter reports showing long-term trends.

Monitoring for the future

When performance is acceptable, you should establish procedures to monitor system performance measurements and anticipate performance constraints before they become response-time problems. Exception-reporting procedures are a key to an effective monitoring approach.

In a complex production system there is usually too much performance data for it to be comprehensively reviewed every day. Key components of performance degradation can be identified with experience, and those components are the ones to monitor most closely. You should identify trends of usage and other factors (such as batch schedules) to aid in this process.

Consistency of monitoring is also important. Just because performance is good for six months after a system is tuned is no guarantee that it will be good in the seventh month.

Reviewing performance data

The aims of the review procedure are to provide continuous monitoring, and to have a good level of detailed data always available so that there is minimal delay in problem analysis.

Generally, there should be a progressive review of data. You should review daily data weekly, and weekly data monthly, unless any incident report or review raises questions that require an immediate check of the next level of detail. This should be enough to detect out-of-line situations with a minimum of effort.

The review procedure also ensures that additional data is available for problem determination, should it be needed. The weekly review should require approximately one hour, particularly after experience has been gained in the process and after you are able to highlight the items that require special consideration. The monthly review will probably take half a day at first. After the procedure has been in force for a period, it will probably be completed more quickly. However, when new applications are installed or when the transaction volumes or numbers of terminals are increased, the process is likely to take longer.

Review the data from the RMF listings only if there is evidence of a problem from the gross-level data, or if there is an end-user problem that can't be solved by the review process. Thus, the only time that needs to be allocated regularly to the detailed data is the time required to ensure that the measurements were correctly made and reported.

When reviewing performance data, try to:

- Establish the basic pattern in the workload of the installation
- Identify variations from the pattern.

Do not discard ***all*** the data you collect, after a certain period. Discard most, but leave a representative sample. For example, do not throw away ***all*** weekly reports after three months; it is better to save those dealing with the last week of each month. At the end of the year, you can discard all except the last week of each quarter. At the end of the following year, you can discard all the previous year's data except for the midsummer week. Similarly, you should keep a representative selection of daily figures and monthly figures.

The intention is that you can compare any report for a ***current*** day, week, or month with an ***equivalent*** sample, however far back you want to go. The samples become more widely spaced but do not cease.

Confirming that the system-oriented objectives are reasonable

After the system is initialized and monitoring is operational, you need to find out if the objectives themselves are reasonable (that is, achievable, given the hardware available), based upon actual measurements of the workload.

When you measure performance against objectives and report the results to users, you have to identify any systematic differences between the measured data and what the user sees. This means an investigation of the differences between internal (as seen by CICS) and external (as seen by the end user) measures of response time.

If the measurements differ greatly from the estimates, you must revise application response-time objectives or plan a reduced application workload, or upgrade your system. If the difference is not too large, however, you can embark on tuning the total system. Parts 3 and 4 of this book tell you how to do this tuning activity.

Typical review questions

Use the following questions as a basis for your own checklist. Most of these questions are answered by the TIVOLI Performance Reporter for OS/390.

Some of the questions are not strictly to do with performance. For instance, if the transaction statistics show a high frequency of transaction abends with usage of the abnormal condition program, this could perhaps indicate signon errors and, therefore, a lack of terminal operator training. This, in itself, is not a performance problem, but is an example of the additional information that can be provided by monitoring.

1. How frequently is each available function used?
   a. Has the usage of transaction identifiers altered?
   b. Does the mix vary from one time of the day to another?
   c. Should statistics be requested more frequently during the day to verify this?

   A different approach must be taken:
   - In systems where all messages are channeled through the same initial task and program (for user security routines, initial editing or formatting, statistical analysis, and so on)
   - For conversational transactions, where a long series of message pairs is reflected by a single transaction
   - In transactions where the amount of work done relies heavily on the input data.

   In these cases, you have to identify the function by program or data set usage, with appropriate reference to the CICS program statistics, file statistics, or other statistics. In addition, you may be able to put user tags into the monitoring data (for example, a user character field in the case of the CICS monitoring facility), which can be used as a basis for analysis by products such as the TIVOLI Performance Reporter.

   The questions asked above should be directed at the appropriate set of statistics.
2. What is the usage of the telecommunication lines?
   a. Do the CICS terminal statistics indicate any increase in the number of messages on the terminals on each of the lines?
   b. Does the average message length on the CICS performance class monitor reports vary for any transaction type? This can easily happen with an application where the number of lines or fields output depends on the input data.
   c. Is the number of terminal errors acceptable? If you are using a terminal error program or node error program, does this indicate any line problems? If not, this may be a pointer to terminal operator difficulties in using the system.
3. What is the DASD usage?
   a. Is the number of requests to file control increasing? Remember that CICS records the number of logical requests made. The number of physical I/Os

depends on the configuration of indexes, and on the data records per control interval and the buffer allocations.

b. Is intrapartition transient data usage increasing? Transient data involves a number of I/Os depending on the queue mix. You should at least review the number of requests made to see how it compares with previous runs.

c. Is auxiliary temporary storage usage increasing? Temporary storage uses control interval access, but writes the control interval out only at syncpoint or when the buffer is full.

4. What is the virtual storage usage?
   a. How large are the dynamic storage areas?
   b. Is the number of GETMAIN requests consistent with the number and types of tasks?
   c. Is the short-on-storage (SOS) condition being reached often?
   d. Have any incidents been reported of tasks being purged after deadlock timeout interval (DTIMOUT) expiry?
   e. How much program loading activity is there?
   f. From the monitor report data, is the use of dynamic storage by task type as expected?
   g. Is storage usage similar at each execution of CICS?
   h. Are there any incident reports showing that the first invocation of a function takes a lot longer than subsequent ones? This may arise when programs are loaded that then have to open data sets, particularly in IMS/ESA, for example. Can this be reconciled with application design?
5. What is the processor usage?
   a. Is the processor usage as measured by the monitor report consistent with previous observations?
   b. Are batch jobs that are planned to run, able to run successfully?
   c. Is there any increase in usage of functions running at a higher priority than CICS? Include in this MVS readers and writers, MVS JES, and VTAM if running above CICS, and overall I/O, because of the lower-priority regions.
6. What is the coupling facility usage?
   a. What is the average storage usage?
   b. What is the ISC link utilization?
7. Do any figures indicate design, coding, or operational errors?
   a. Are any of the resources mentioned above heavily used? If so, was this expected at design time? If not, can the heavy use be explained in terms of heavier use of transactions?
   b. Is the heavy usage associated with a particular application? If so, is there evidence of planned growth or peak periods?
   c. Are browse transactions issuing more than the expected number of requests? In other words, is the count of browse requests issued by a transaction greater than what you expected users to cause?
   d. Is the CICS CSAC transaction (provided by the DFHACP abnormal condition program) being used frequently? Is this because invalid transaction identifiers are being entered? For example, errors are signaled if transaction identifiers are entered in lowercase on IBM® 3270 terminals but automatic translation of input to uppercase has not been specified.

      A high use of the DFHACP program without a corresponding count of CSAC may indicate that transactions are being entered without proper

operator signon. This may, in turn, indicate that some terminal operators need more training in using the system.

In addition to the above, you should regularly review certain items in the CICS statistics, such as:

- Times the MAXTASK limit reached (transaction manager statistics)
- Peak tasks (transaction class statistics)
- Times cushion released (storage manager statistics)
- Storage violations (storage manager statistics)
- Maximum RPLs posted (VTAM statistics)
- Short-on-storage count (storage manager statistics)
- Wait on string total (file control statistics)
- Use of DFHSHUNT log streams.
- Times aux. storage exhausted (temporary storage statistics)
- Buffer waits (temporary storage statistics)
- Times string wait occurred (temporary storage statistics)
- Times NOSPACE occurred (transient data global statistics)
- Intrapartition buffer waits (transient data global statistics)
- Intrapartition string waits (transient data global statistics)

You should also satisfy yourself that large numbers of dumps are not being produced.

Furthermore, you should review the effects of and reasons for system outages and their duration. If there is a series of outages, you may be able to detect a common cause of them.

Anticipating and monitoring system changes and growth

No production system is static. Each system is constantly changing because of new function being added, increased transaction volumes because of a growth in the number of terminal users, addition of new applications or software components, and changes to other aspects of the data processing complex (batch, TSO, and so on). As much as possible, the effects of these changes need to be anticipated, planned for, and monitored.

To find out what application changes are planned, interviewing system or application development managers can be useful in determining the effect of new function or applications and the timing of those changes. Associated with this is the effect of new software to be installed, as well as the known hardware plans for installing new equipment.

When a major change to the system is planned, increase the monitoring frequency before and after the change. A major change includes the addition of:

- A new application or new transactions
- New terminals
- New software releases.

You should look at individual single-thread transactions as well as the overall behavior of the production system.

If the system performance has altered as a result of a major change to the system, data for before-and-after comparison of the appropriate statistics provides the best way of identifying the reasons for the alteration.

Consider having extra tools installed to make it easier to project and test future usage of the system. Tools such as the Teleprocessing Network Simulator (TPNS) program can be used to test new functions under volume conditions before they actually encounter production volumes. Procedures such as these can provide you with insight as to the likely performance of the production system when the changes are implemented, and enable you to plan option changes, equipment changes, scheduling changes, and other methods for stopping a performance problem from arising.

Part 2. Tools that measure the performance of CICS

This part gives an overview of the various tools that can be used to find out which resources are in contention.

Chapter 4. An overview of performance-measurement tools

This overview discusses methods of measuring performance in the following sections:

After reasonable performance objectives have been agreed, you have to set up methods to determine whether the production system is meeting those objectives.

Performance of a production system depends on the utilization of resources such as CPU, real storage, ISC links, coupling facility, and the network.

You have to monitor all of these factors to determine when constraints in the system may develop. A variety of programs could be written to monitor all these resources. Many of these programs are currently supplied as part of IBM products such as CICS or IMS/ESA, or are supplied as separate products. This chapter describes some of the products that can give performance information on different components of a production system.

The list of products in this chapter is far from being an exhaustive summary of performance monitoring tools, yet the data provided from these sources comprises a large amount of information. To monitor all this data is an extensive task. Furthermore, only a small subset of the information provided is important for identifying constraints and determining necessary tuning actions, and you have to identify this specific subset for your particular CICS system.

You also have to bear in mind that there are two different types of tools:

1. Tools that directly measure whether you are meeting your objectives
2. Additional tools to look into internal reasons why you might not be meeting objectives.

None of the tools can directly measure whether you are meeting end-user response time objectives. The lifetime of a task within CICS is comparable, that is, usually related to, response time, and bad response time is usually correlated with long lifetime within CICS, but this correlation is not exact because of other contributors to response time.

Obviously, you want tools that help you to measure your objectives. In some cases, you may choose a tool that looks at some internal function that contributes towards your performance objectives, such as task lifetime, rather than directly measuring the actual objective, because of the difficulty of measuring it.

When you have gained experience of the system, you should have a good idea of the particular things that are most significant in that particular system and, therefore, what things might be used as the basis for exception reporting. Then, one way of simply monitoring the important data might be to set up exception-reporting procedures that filter out the data that is not essential to the tuning process. This involves setting standards for performance criteria that identify constraints, so that the exceptions can be distinguished and reported while

normal performance data is filtered out. These standards vary according to individual system requirements and service level agreements.

You often have to gather a considerable amount of data before you can fully understand the behavior of your own system and determine where a tuning effort can provide the best overall performance improvement. Familiarity with the analysis tools and the data they provide is basic to any successful tuning effort.

Remember, however, that all monitoring tools cost processing effort to use. Typical costs are 5% additional processor cycles for the CICS monitoring facility (performance class), and up to 1% for the exception class. The CICS trace facility overhead is highly dependent on the workload used. The overhead can be in excess of 25%.

In general, then, we recommend that you use the following tools in the sequence of priorities shown below:

1. CICS statistics
2. CICS monitoring data
3. CICS internal and auxiliary trace.

In this chapter, the overview of the various tools for gathering or analyzing data is arranged as follows:

- *CICS performance data*
- *Operating system performance data*
- *Performance data for other products*.

CICS performance data

CICS statistics

CICS statistics are the simplest and the most important tool for permanently monitoring a CICS system. They collect information on the CICS system as a whole, without regard to tasks.

The CICS statistics domain writes five types of statistics to SMF data sets: *interval, end-of-day, requested, requested reset*, and *unsolicited* statistics.

Each of these sets of data is described and a more general description of CICS statistics is given in "Chapter 5. Using CICS statistics" on page 35and "Appendix A. CICS statistics tables" on page 339.

The CICS monitoring facility

The CICS monitoring facility collects information about CICS tasks, and is described more completely in "Chapter 6. The CICS monitoring facility" on page 61.

The *CICS Customization Guide* contains programming information on the data set formats and the *CICS Operations and Utilities Guide* describes the monitoring utility programs, DFHMNDUP and DFH$MOLS.

The sample statistics program (DFH0STAT)

You can use the statistics sample program, DFH0STAT, to help you determine and adjust the values needed for CICS storage parameters, for example, using DSALIM and EDSALIM. The program produces a report showing critical system parameters from the CICS dispatcher, an analysis of the CICS storage manager and loader statistics, and an overview of the MVS storage in use. The program demonstrates the use of the EXEC CICS INQUIRE and EXEC CICS COLLECT STATISTICS commands to produce an analysis of a CICS system. You can use the sample program as provided or modify it to suit your needs. It can be used to provide data about the following:

- System Status, Monitoring and Statistics
- Transaction Manager and Dispatcher
- Storage
- Loader
- Storage Subpools
- Transaction Classes
- Transactions
- Transaction Totals including Subspace usage information
- Programs
- Program Totals
- DFHRPL Analysis
- Programs by DSA and LPA
- Temporary Storage
- Temporary Storage Queues
- Temporary Storage Queues by TSPOOL
- Transient Data
- Transient Data Queues
- Transient Data Queues Total
- User Domain
- Journalnames
- Logstreams
- Connections and Modenames
- TCP/IP Services
- Autoinstall and VTAM
- LSR Pools
- Files
- Coupling Facility Data Table Pools
- DB2® Connections and Entries
- Data Tables
- Exit Programs
- Global User Exits
- Enqueue Manager
- Recovery Manager.

See "Appendix E. The sample statistics program, DFH0STAT" on page 513 for the details and interpretation of the report.

CICS trace facilities

For the more complex problems that involve system interactions, you can use the CICS trace to record the progress of CICS transactions through the CICS management modules. Whereas a dump gives a "snapshot" of conditions at a particular moment, CICS trace provides a history of events leading up to a specific situation. CICS includes facilities for selective activation or deactivation of some groups of traces.

The CICS trace facilities can also be useful for analyzing performance problems such as excessive waiting on events in the system, or constraints resulting from inefficient system setup or application program design.

Several types of tracing are provided by CICS, and are described in the *CICS Problem Determination Guide*. Trace is controlled by:

- The system initialization parameters (see the *CICS System Definition Guide*).
- CETR (see the *CICS Supplied Transactions* manual). CETR also provides for trace selectivity by, for instance, transaction type or terminal name.
- CEMT SET INTTRACE, CEMT SET AUXTRACE, or CEMT SET GTFTRACE (see the *CICS Supplied Transactions* manual).
- EXEC CICS SET TRACEDEST, EXEC CICS SET TRACEFLAG, or EXEC CICS SET TRACETYPE (see the *CICS System Programming Reference* for programming information).

Three destinations are available for trace data:

1. The internal trace table, in main storage above the 16MB line
2. Auxiliary trace data sets, defined as BSAM data sets on tape or disk
3. The MVS generalized trace facility (GTF) data sets, which can be accessed through the MVS interactive problem control system (IPCS).

Other CICS data

The measurement tools previously described do not provide all the data necessary for a complete evaluation of current system performance. They do not provide information on how and under what conditions each resource is being used, nor do they provide information about the existing system configuration while the data is being collected. It is therefore extremely important to use as many techniques as possible to get information about the system. Additional sources of information include the following:

- Hardware configuration
- VTOC listings
- LISTCAT (VSAM)
- CICS table listings, especially:
  - SIT (and overrides in the CICS startup procedure)
  - FCT (file control table)for any BDAM files
- CICS resource definitions from the CSD file:
  - Use the DFHCSDUP LIST command to print resource definitions, groups, and lists. For information about the CSD file utility program, DFHCSDUP, see the *CICS Resource Definition Guide*.
- Link pack area (LPA) map
- Load module cross-reference of the CICS nucleus
- SYS1.PARMLIB listing

- MVS Workload Manager (WLM) service definition
- MVS System Logger configuration - LOGR couple data set listing
- Dump of the CICS address space. See the CICS Operations and Utilities Guide for information on how to get an address space dump for CICS when the CICS address space abends.

This data, used with the data produced by the measurement tools, provides the basic information that you should have for evaluating your system's performance.

Operating system performance data

- "System management facility (SMF)"
- "Resource measurement facility (RMF)".
- "Generalized trace facility (GTF)" on page 27
- "Tivoli Performance Reporter for OS/390" on page 29

System management facility (SMF)

System management facilities (SMF) collects and records system and job-related information that your installation can use in:

- Billing users
- Reporting reliability
- Analyzing the configuration
- Scheduling jobs
- Summarizing direct access volume activity
- Evaluating data set activity
- Profiling system resource use
- Maintaining system security.

For more information on SMF, see the *OS/390 MVS System Management Facilities (SMF)* manual, GC28-1783-05.

Resource measurement facility (RMF)

The Resource Measurement Facility (RMF) collects system-wide data that describes the processor activity (WAIT time), I/O activity (channel and device usage), main storage activity (demand and swap paging statistics), and system resources manager (SRM) activity (workload).

RMF is a centralized measurement tool that monitors system activity to collect performance and capacity planning data. The analysis of RMF reports provides the basis for tuning the system to user requirements. They can also be used to track resource usage.

RMF measures the following activities:

- Processor usage
- Address space usage
- Channel activity:
  - Request rate and service time per physical channel
  - Logical-to-physical channel relationships
  - Logical channel queue depths and reasons for queuing.
- Device activity and contention for the following devices:

  - Unit record
  - Graphics
  - Direct access storage
  - Communication equipment
  - Magnetic tapes
  - Character readers.
- Detailed system paging
- Detailed system workload
- Page and swap data set
- Enqueue
- CF activity
- XCF activity.

RMF allows the OS/390 user to:

- Evaluate system responsiveness:
  - Identify bottlenecks. The detailed paging report associated with the page and swap data set activity can give a good picture of the behavior of a virtual storage environment.
- Check the effects of tuning:
  - Results can be observed dynamically on a screen or by postprocessing facilities.
- Perform capacity planning evaluation:
  - The workload activity reports include the interval service broken down by key elements such as processor, input/output, and main storage service.
  - Analysis of the resource monitor output (for example, system contention indicators, swap-out broken down by category, average ready users per domain) helps in understanding user environments and forecasting trends.
  - The post-processing capabilities make the analysis of peak load periods and trend analysis easier.
- Manage the larger workloads and increased resources that MVS can support
- Identify and measure the usage of online channel paths
- Optimize the usefulness of expanded storage capability.

RMF measures and reports system activity and, in most cases, uses a sampling technique to collect data. Reporting can be done with one of three monitors:

1. Monitor I measures and reports the use of system resources (that is, the processor, I/O devices, storage, and data sets on which a job can enqueue during its execution). It runs in the background and measures data over a period of time. Reports can be printed immediately after the end of the measurement interval, or the data can be stored in SMF records and printed later with the RMF postprocessor. The RMF postprocessor can be used to generate reports for "exceptions": conditions where user-specified values are exceeded.
2. Monitor II, like Monitor I, measures and reports the use of system resources. It runs in the background under TSO or on a console. It provides "snapshot" reports about resource usage, and also allows its data to be stored in SMF records. The RMF postprocessor can be used to generate exception reports.
3. Monitor III primarily measures the contention for system resources and the delay of jobs that such contention causes. It collects and reports the data in real time at a display station, with optional printed copy backup of individual

displays. Monitor III can also provide exception reports, but its data ***cannot*** be stored in SMF records. It must be used if XCF or CF reports are needed.

RMF should be active in the system 24 hours a day, and you should run it at a dispatching priority above other address spaces in the system so that:

- The reports are written at the interval requested
- Other work is not delayed because of locks held by RMF.

A report is generated at the time interval specified by the installation. The largest system overhead of RMF occurs during the report generation: the shorter the interval between reports, the larger the burden on the system. An interval of 60 minutes is recommended for normal operation. When you are addressing a specific problem, reduce the time interval to 10 or 15 minutes. The RMF records can be directed to the SMF data sets with the NOREPORT and RECORD options; the report overhead is not incurred and the SMF records can be formatted later.

Note: There may be some discrepancy between the CICS initialization and termination times when comparing RMF reports against output from the CICS monitoring facility.

For further details of RMF, see the *OS/390 Resource Measurement Facility (RMF) Users Guide*, *SC28-1949*.

Guidance on how to use RMF with the CICS monitoring facility is given in "Using CICS monitoring SYSEVENT information with RMF" on page 63. In terms of CPU costs this is an inexpensive way to collect performance information. Shorter reports throughout the day are needed for RMF because a report of a full day's length includes startup and shutdown and does not identify the peak period.

Generalized trace facility (GTF)

As described above, CICS trace entries can be recorded via GTF, and reports produced via IPCS. More generally, GTF is an integral part of the MVS system, and traces the following system events: DASD seek addresses on start I/O instructions, system resources manager (SRM) activity, page faults, I/O activity, and supervisor services. Execution options specify the system events to be traced. The amount of processing time to be used by GTF can vary considerably, depending on the number of events to be traced. You should request the time-stamping of GTF records with the TIME=YES operand on the EXEC statement for all GTF tracing.

GTF should run at a dispatching priority (DPRTY) of 255 so that records are not lost. If GTF records are lost and the DPRTY is specified at 255, specify the BUF operand on the execute statement as greater than 10 buffers.

GTF is generally used to monitor short periods of system activity and you should run it accordingly.

You can use these options to get the data normally needed for CICS performance studies:

```
TRACE=SYS,RNIO,USR     (VTAM)
TRACE=SYS              (Non-VTAM)
```

If you need data on the units of work dispatched by the system and on the length of time it takes to execute events such as SVCs, LOADs, and so on, the options are:

```
TRACE=SYS,SRM,DSP,TRC,PCI,USR,RNIO
```

The TRC option produces the GTF trace records that indicate GTF interrupts of other tasks that it is tracing. This set of options uses a higher percentage of processor resources, and you should use it only when you need a detailed analysis or timing of events.

No data-reduction programs are provided with GTF. To extract and summarize the data into a meaningful and manageable form, you can either write a data-reduction program or use one of the program offerings that are available.

For further details, see the *OS/390 MVS Diagnosis: Tools and Service Aids*.

GTF reports

You can produce reports from GTF data using the interactive problem control system (IPCS). The reports generated by IPCS are useful in evaluating both system and individual job performance. It produces job and system summary reports as well as an abbreviated detail trace report. The summary reports include information on MVS dispatches, SVC usage, contents supervision, I/O counts and timing, seek analysis, page faults, and other events traced by GTF. The detail trace reports can be used to follow a transaction chronologically through the system.

Other reports are available that:

- Map the seek addresses for a specific volume
- Map the arm movement for a specific volume
- Map the references to data sets and members within partitioned data sets
- Map the page faults and module reference in the link pack area (LPA).

These reports are described later in this section.

Before GTF is run, you should plan the events to be traced. If specific events such as start I/Os (SIOs) are not traced, and the SIO-I/O timings are required, the trace must be re-created to get the data needed for the reports.

If there are any alternative paths to a control unit in the system being monitored, you should include the PATHIO input statement in the report execution statement. Without the PATHIO operand, there are multiple I/O lines on the report for the device with an alternative path: one line for the primary device address and one for the secondary device address. If this operand is not included, the I/Os for the primary and alternate device addresses have to be combined manually to get the totals for that device.

Seek histogram report

The seek histogram report (SKHST) can help you find out if there is any arm contention on that volume, that is, if there are any long seeks on the volume being mapped. It produces two reports: the first shows the number of seeks to a particular address, and the second shows the distance the arm moves between seeks. These reports can be used to determine if you should request a volume map report to investigate further the need to reorganize a specific volume.

Volume map report

The volume map report (VOLMAP) displays information about data sets on the volume being mapped and about seek activity to each data set on that volume. It also maps the members of a partitioned data set and the count of seeks issued to each member. This report can be very useful in reorganizing the data sets on a

volume and in reorganizing the members within a partitioned data set to reduce the arm movement on that specific volume.

Reference map report

The reference map report (REFMAP) shows the page fault activity in the link pack area (LPA) of MVS. This reference is by module name and separates the data faults from the instruction faults. The report also shows the count of references to the specific module. This reference is selected from the address in the stored PSW of the I/O and EXT interrupt trace events from GTF. This report can be useful if you want to make changes to the current MVS pack list in order to reduce real storage or to reduce the number of page faults that are being encountered in the pageable link pack area of MVS.

Tivoli Performance Reporter for OS/390

Tivoli Performance Reporter for OS/390 is an IBM product that collects and analyzes data from CICS and other IBM systems and products. With the Tivoli Performance Reporter you can build reports which help you with the following:

- System overviews
- Service levels
- Availability
- Performance and tuning
- Capacity planning
- Change and problem management
- Accounting.

A large number of ready-made reports are available, and in addition you can generate your own reports to meet specific needs.

In the reports the Tivoli Performance Reporter uses data from CICS monitoring and statistics. Tivoli Performance Reporter also collects data from the MVS system and from products such as RMF, TSO, IMS™ and NetView. This means that data from CICS and other systems can be shown together, or can be presented in separate reports.

Reports can be presented as plots, bar charts, pie charts, tower charts, histograms, surface charts, and other graphic formats. The Tivoli Performance Reporter for OS/390 simply passes the data and formatting details to Graphic Data Display Manager (GDDM) which does the rest. The Tivoli Performance Reporter can also produce line graphs and histograms using character graphics where GDDM is not available, or the output device does not support graphics. For some reports, where you need the exact figures, numeric reports such as tables and matrices are more suitable.

See “Chapter 7. Tivoli Performance Reporter for OS/390” on page 109 for more information about the Tivoli Performance Reporter for OS/390 as a CICS performance measurement tool.

Performance data for other products

This section gives an overview of the tools that can be used to monitor information on various access methods and other programs used with CICS and the operating system.

ACF/VTAM

ACF/VTAM® (program number 5735-RC2) provides information about buffer usage either to GTF in SMF trace data or to the system console through DISPLAY and BFRUSE commands. Other tuning statistics can also be recorded on the system console through the MODIFY procname, TNSTAT command. (This command is described in the *ACF/VTAM Diagnostic Techniques* manual.)

Virtual telecommunication access method (VTAM) trace

The VTAM trace facility is provided as part of VTAM, and tracks messages through different points to and from CICS. The time-stamps that are included can be particularly useful in determining where a transaction spends large amounts of time.

Network performance, analysis, and reporting system (NETPARS)

NETPARS is a program offering (program number 5798-CZX) that analyzes network log data from the NetView® Performance Monitor (NPM). Further information on NETPARS is given in the *Network Performance, Analysis, and Reporting System (NETPARS) Description/Operations*.

VTAM performance, analysis, and reporting system II (VTAMPARS II)

The VTAMPARS program offering (program number 5798-DFE) provides information on network traffic through the VTAM component of a network. Information on terminal connect time, message characteristics and rates, and so forth, can be collected and analyzed. Further information on VTAMPARS is given in the *VTAM Performance, Analysis, and Reporting System (VTAMPARS) Program Description/Operations*.

Generalized performance analysis reporting (GPAR)

Generalized Performance Analysis Reporting (GPAR) (program number 5798-CPR) is a prerequisite for VTAMPARS. GPAR is designed as a base for reporting programs (IBM or user-written). It helps summarize sequential activity traces like GTF traces. It also contains facilities to print user-tailored graphs from any performance data log or non-VSAM sequential data set.

VTAM storage management (SMS) trace

The VTAM storage management (SMS) trace facility collects information on VTAM's usage of its buffers, including which buffers are used in the various buffer pools, and the number of buffer expansions and depletions.

VTAM tuning statistics

Information provided in the VTAM tuning statistics includes data on the performance between VTAM and the network control program (NCP), the number of reads and writes and what caused that activity, and message counts.

NetView for MVS

NetView is a network management program offering (program number 5665-362) which provides a cohesive set of SNA host network management services in a single product. NetView includes the functions of the network communication control facility (NCCF), network logical data manager (NLDM), and network problem determination application (NPDA), as well as functions of the VTAM node control application (VNCA) and network management productivity facility (NMPF). Support is provided for problem determination of the IBM 3720 Communication Controller and for online configuration control and testing of IBM 586X modems.

NetView's set of network management functions consists of the following:

- Command facility
- Session monitor
- Hardware monitor
- Status monitor
- Online HELP and Help Desk facility
- Browse facility.

NetView's capabilities include:

- Terminal access facility support of large screen and color applications (for example, the NetView performance monitor)
- CLISTs driven by application messages
- Disk log enhancements
- Support for 586X Models 2 and 3 and 5812 modems
- Token-ring network support
- Virtual route blockage indication
- Session setup failure notification
- Extended recovery facility
- Automatic operations and recovery
- Real-time update of the domain status panel
- Easy-to-use installation procedure.

The benefits provided by NetView include:

- Improved cohesion and usability in support of network management functions
- Enhanced installation, operation, and utilization of network management functions in MVS environments.

For further information on NetView, see the *Systems Network Architecture Management Services Reference*, and *Network Program Products Planning* manual.

NetView performance monitor (NPM)

The NetView Performance Monitor (NPM) program product (program number 5665-333) is designed to aid network support personnel in managing VTAM-based communications networks. It collects and reports on data in the host and NCP.

NPM data can be used to:

- Identify network traffic bottlenecks
- Display screens showing volume and response times for various resources
- Generate color graphs of real-time and historical data
- Alert users to response time threshold exceptions.

NPM performance data can also help to:

- Determine the performance characteristics of a network and its components
- Identify network performance problems
- Tune communications networks for better performance as well as verify the effects of problem resolutions
- Gauge unused capacity when planning for current network changes
- Produce timely and meaningful reports on network status for multiple levels of management.

Further information on NPM is given in *NetView Performance Monitor At A Glance*.

LISTCAT (VSAM)

VSAM LISTCAT provides information that interprets the actual situation of VSAM data sets. This information includes counts of the following:

- Whether and how often control interval (CI) or control area (CA) splits occur (splits should occur very rarely, especially in CA).
- Physical accesses to the data set.
- Extents for a data set (secondary allocation). You should avoid this secondary allocation, if possible, by making the primary allocation sufficiently large.
- Index levels.

Virtual storage access method (VSAM) or ICF catalog

Information kept in the VSAM or Integrated Catalog Facility (ICF) catalog includes items on record sizes, data set activity, and data set organization.

DB monitor (IMS)

The IMS DB monitor report print program (DFSUTR30) provides information on batch activity (a single-thread environment) to IMS databases, and is activated through the DLMON system initialization parameter. As in the case of CICS auxiliary trace, this is for more in-depth investigation of performance problems by single-thread studies of individual transactions.

The DB monitor cannot be started and stopped from a terminal. After the DB monitor is started in a CICS environment, the only way to stop it is to shut down CICS. The DB monitor cannot be started or stopped dynamically.

When the DB monitor runs out of space on the IMSMON data set, it stops recording. The IMSMON data set is a sequential data set, for which you can allocate space with IEFBR14. The DCB attributes are:

```
DCB=(RECFM=VB,LRECL=2044,BLKSIZE=2048)
```

If you are running the DB monitor in a multithread (more than one) environment, the only statistics that are valid are the VSAM buffer pool statistics.

Program isolation (PI) trace

The program isolation (PI) trace can point out database contention problems arising from the nature of task's access to a particular database. Because only one task can have access to a record at one time, and any other task waits till the record is freed, high contention can mean high response time. This trace is part of IMS, and can be activated by the CEMT SET PITRACE ON|OFF command. Information on the format of the PI trace report is given in the *IMS/ESA Version 3 System Administration Guide*.

IMS System Utilities/Database Tools (DBT)

The IMS System Utilities/Database Tools (DBT) program product (program number 5668-856) is a powerful package of database programs and products designed to enhance data integrity, data availability, and performance of IMS databases. It provides the important tools that are needed to support both full-function (index, HDAM, HIDAM, and HISAM) and fastpath (DEDB) databases.

DBT can help you maintain data integrity by assisting the detection and repair of errors before a problem disrupts operations. It speeds database reorganization by providing a clear picture of how data is stored in the database, by allowing the user to simulate various database designs before creating a new database, and by providing various sort, unload, and reload facilities. DBT also improves programming productivity by providing monitoring capabilities and by reducing the need to write reformatting programs. It increases the user's understanding of the database for analysis, tuning, and reorganization. It also helps enhance the overall database performance.

For further information, see the *IMS System Utilities/Database Tools (DBT) General Information* manual.

IMS monitor summary and system analysis II (IMSASAP II)

IMSASAP II (program number 5798-CHJ) is a performance analysis and tuning aid for IMS/ESA® database and data communication systems. It is a report program that executes under IMS/ESA for Generalized Performance Analysis Reporting (GPAR). IMSASAP II processes IMS/ESA DB and DC monitor data to provide summary, system analysis, and program analysis level reports that assist in the analysis of an IMS/ESA system environment. The monitor concept has proven to be a valuable aid in the performance analysis and tuning of IMS systems. IMSASAP II extends this capability by providing comprehensive reports (from management summaries to detail program traces) to meet a broad range of IMS/ESA system analysis objectives.

IMSASAP:

- Produces a comprehensive set of reports, organized by level of detail and area of analysis, to satisfy a wide range of IMS/ESA system analysis requirements
- Provides report selection and reporting options to satisfy individual requirements and to assist in efficient analysis
- Produces alphanumerically collated report items in terms of ratios, rates, and percentages to facilitate a comparison of results without additional computations
- Reports on schedules in progress including wait-for-input and batch message processing programs
- Provides reports on IMS/ESA batch programs.

Further information on IMSASAP is given in the *IMSASAP II Program Description/Operations* manual.

DATABASE 2 Performance Monitor (DB2PM)

DATABASE 2™ Performance Monitor (program number 5665-354) analyses DB2 performance data and generates a comprehensive set of reports. These include the following:

- A set of graphs showing DB2 statistics, accounting, and frequency distribution performance data
- A summary of DB2 system activity, including system tasks (statistics data)
- A summary of DB2 application work, reported either by user or by application (accounting data)
- A set of transit time reports detailing DB2 workload performance
- System- and application-related DB2 I/O activity
- Locking activity, reported both by DB2 application type and by database
- SQL activity
- Selective tracing and formatting of DB2 records.

For further information, see the *DATABASE 2 Performance Monitor (DB2PM) General Information* manual.

Teleprocessing network simulator (TPNS)

The Teleprocessing Network Simulator (TPNS) (program number 5662-262) is a program that simulates terminal activity such as that coming through the NCP. TPNS can be used to operate an online system at different transaction rates, and can monitor system performance at those rates. TPNS also keeps information on response times, which can be analyzed after a simulation.

Further information on TPNS is given in the *Teleprocessing Network Simulator (TPNS) General Information* manual.

Chapter 5. Using CICS statistics

This chapter discusses CICS statistics in the following sections: Methods for collecting statistics are described, and statistics that can be used for tuning your CICS system are included.

Introduction to CICS statistics

CICS management modules control how events are managed by CICS. As events occur, CICS produces information that is available to you as system and resource statistics.

The resources controlled by CICS include files, databases, journals, transactions, programs, and tasks. Resources that CICS manages, and values that CICS uses in its record-keeping role, are defined in one of the following ways:

- Online, by the CICS CEDA transaction.
- Offline, by the CICS system definition (CSD) utility program, DFHCSDUP. See the *CICS Customization Guide* for programming information about DFHCSDUP.
- Offline, by CICS control table macros.

Statistics are collected during CICS online processing for later offline analysis. The statistics domain writes statistics records to a System Management Facilities (SMF) data set. The records are of SMF type 110, sub-type 002. Monitoring records and some journaling records are also written to the SMF data set as type 110 records. You might find it useful to process statistics and monitoring records together. For programming information about SMF, and about other SMF data set considerations, see the *CICS Customization Guide*.

Types of statistics data

CICS produces five types of statistics:

Interval statistics

Are gathered by CICS during a specified interval. CICS writes the interval statistics to the SMF data set automatically at the expiry of the interval if:

- Statistics recording status was set ON by the STATRCD system initialization parameter (and has not subsequently been set OFF by a CEMT or EXEC CICS SET STATISTICS RECORDING command). The default is STATRCD=OFF.
- ON is specified in CEMT SET STATISTICS.
- The RECORDING option of the EXEC CICS SET STATISTICS command is set to ON.

End-of-day statistics

Are a special case of interval statistics where all statistics counters are collected and reset. There are three ways to get end-of-day statistics:

- The end-of-day expiry time
- When CICS quiesces (normal shutdown)

- When CICS terminates (immediate shutdown).

The end of day value defines a logical point in the 24 hour operation of CICS. You can change the end of day value using CEMT SET STATISTICS or the EXEC CICS SET STATISTICS command. End-of-day statistics are always written to the SMF data set, regardless of the settings of any of the following:

- The system initialization parameter, STATRCD, or
- CEMT SET STATISTICS or
- The RECORDING option of EXEC CICS SET STATISTICS.

The statistics that are written to the SMF data set are those collected since the last event which involved a reset. The following are examples of resets:

- At CICS startup
- Issue of RESETNOW RECORDNOW in CEMT or EXEC CICS STATISTICS commands.
- Interval statistics

The default end-of-day value is 000000 (midnight).

End-of-day statistics are always written to the SMF data set, regardless of the settings of any of the following:

- The system initialization parameter, STATRCD, or
- CEMT SET STATISTICS or
- The RECORDING option of EXEC CICS SET STATISTICS.

Requested statistics
are statistics that the user has asked for by using one of the following commands:

- CEMT PERFORM STATISTICS RECORD
- EXEC CICS PERFORM STATISTICS RECORD
- EXEC CICS SET STATISTICS ON|OFF RECORDNOW.

These commands cause the statistics to be written to the SMF data set immediately, instead of waiting for the current interval to expire. The PERFORM STATISTICS command can be issued with any combination of resource types or you can ask for all resource types with the ALL option. For more details about CEMT commands see the *CICS Supplied Transactions*; for programming information about the equivalent EXEC CICS commands, see the *CICS System Programming Reference*.

Requested reset statistics
differ from requested statistics in that all statistics are collected and statistics counters are reset. You can reset the statistics counters using the following commands:

- CEMT PERFORM STATISTICS RECORD ALL RESETNOW
- EXEC CICS PERFORM STATISTICS RECORD ALL RESETNOW
- EXEC CICS SET STATISTICS ON|OFF RESETNOW RECORDNOW

The PERFORM STATISTICS command must be issued with the ALL option if RESETNOW is present.

You can also invoke requested reset statistics when changing the recording status from ON to OFF, or vice versa, using CEMT SET STATISTICS

ON|OFF RECORDNOW RESETNOW, or EXEC CICS SET STATISTICS ON|OFF RECORDNOW RESETNOW.

Note: It is valid to specify RECORDNOW RESETNOW options only when there is a genuine change of status from STATISTICS ON to OFF, or vice versa. In other words, coding EXEC CICS SET STATISTICS ON RECORDNOW RESETNOW when statistics is already ON will cause an error response.

RESETNOW RECORDNOW on the SET STATISTICS command can only be invoked if the RECORDING option is changed. See also Figure 1.

Note: Issuing the RESETNOW command by itself in the SET STATISTICS command causes the loss of the statistics data that has been collected since the last interval. Interval collections take place only if you set the RECORDING status ON. To set the statistics recording status ON or OFF, use either the RECORDING option on this command or the SIT parameter STATRCD. Statistics are always written, and counts reset, at the end of day. See Figure 1 for further information.

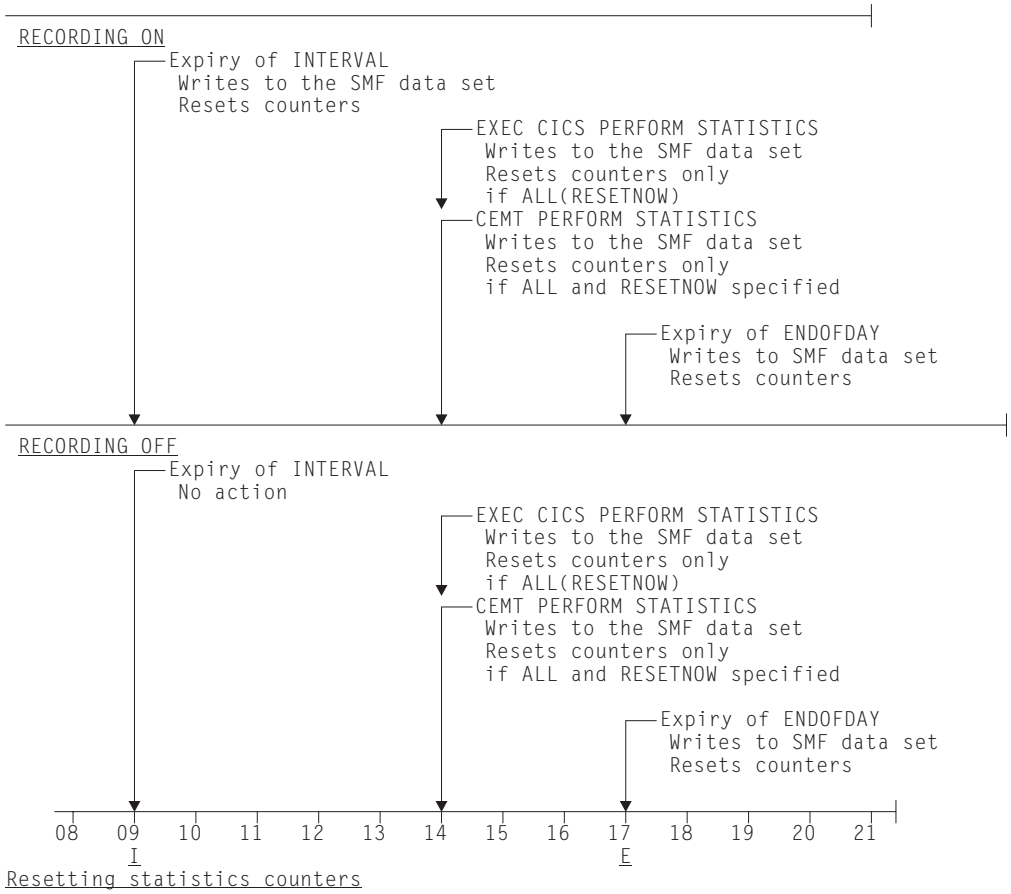


Figure 1. Summary of statistics reset functions

Unsolicited statistics
are automatically gathered by CICS for dynamically allocated and deallocated resources. CICS writes these

statistics to SMF just before the resource is deleted regardless of the status of statistics recording.

Unsolicited statistics are produced for:

autoinstalled terminals
: Whenever an autoinstalled terminal entry in the TCT is deleted (after the terminal logs off), CICS collects statistics covering the autoinstalled period since the last interval. The period covers any delay interval specified by the system initialization parameter, AILDELAY.

  If an autoinstall terminal logs on again before the expiry of the delay interval, the accumulation of statistics continues until the next interval. At that interval, the accumulation of statistics is restarted.

DBCTL
: Whenever CICS disconnects from DBCTL, CICS collects the statistics covering the whole of the DBCTL connection period.

DB2
: Whenever CICS disconnects from DB2, CICS collects the statistics for the DB2 connection and all DB2ENTRYs covering the period from the last interval.

  Whenever a DB2ENTRY is discarded, CICS collects the statistics for that DB2ENTRY covering the period from the last interval.

FEPI connection
: Unsolicited connection statistics are produced when a connection is destroyed. This could occur when a DISCARD TARGET, DISCARD NODE, DISCARD POOL, DELETE POOL, DISCARD NODELIST, or DISCARD TARGETLIST command is used.

FEPI pools
: Unsolicited pool statistics are produced when a pool is discarded by using the DISCARD POOL or DELETE POOL command.

FEPI targets
: Unsolicited target statistics are produced when a target is destroyed or removed from a pool. This occurs when a DELETE POOL, DISCARD POOL, DISCARD TARGET, or DISCARD TARGETLIST command is used.

files
: Whenever CICS closes a file, CICS collects statistics covering the period from the last interval.

JOURNALNAMES
: Unsolicited journalname statistics are produced when a journalname is discarded by using the DISCARD JOURNALNAME command.

LOGSTREAMS
Unsolicited logstream statistics are produced when the logstream is discarded from the MVS system logger.

LSRpools
When CICS closes a file which is in an LSRPOOL, CICS collects the statistics for the LSRPOOL. The following peak values are reset at each interval collection:

- Peak number of requests waiting for a string
- Maximum number of concurrent active file control strings.

The other statistics, which are not reset at an interval collection, cover the entire period from the time the LSRPOOL is created (when the first file is opened) until the LSRPOOL is deleted (when the last file is closed).

PROGRAMS
When an installed program definition is discarded, CICS collects the statistics covering the installed period since the last interval.

TCP/IP Services
Whenever CICS closes a TCP/IP service, CICS collects the statistics covering the period since the last interval.

TRANSACTIONS
When an installed transaction definition is discarded, CICS collects the statistics covering the installed period since the last interval.

TRANSACTION CLASSES
When an installed transaction class definition is discarded, CICS collects the statistics covering the installed period since the last interval.

TRANSIENT DATA QUEUES
Unsolicited transient data queue statistics are produced when a transient data queue is discarded by using DISCARD TDQUEUE, or when an extrapartition transient data queue is closed.

Note: To ensure that accurate statistics are recorded unsolicited statistics (USS) must be collected. An unsolicited record resets the statistics fields it contains. In particular, during a normal CICS shutdown, files are closed before the end of day statistics are gathered. This means that file and LSRPOOL end of day statistics will be zero, while the correct values will be recorded as unsolicited statistics.

Resetting statistics counters

When statistics are written to the SMF data set, the counters are reset in one of the following ways:

- Reset to zero

- Reset to 1
- Reset to current values (this applies to peak values)
- Are not reset
- Exceptions to the above.

For detailed information about the reset characteristics, see "Appendix A. CICS statistics tables" on page 339.

The arrival of the end-of-day time, as set by the ENDOFDAY parameters, always causes the current interval to be ended (possibly prematurely) and a new interval to be started. Only end-of-day statistics are collected at the end-of-day time, even if it coincides exactly with the expiry of an interval.

Changing the end-of-day value changes the times at which INTERVAL statistics are recorded immediately. In Figure 2, when the end-of-day is changed from midnight to 1700 just after 1400, the effect is for the interval times to be calculated from the new end-of-day time. Hence the new interval at 1500 as well as for the times after new end-of-day time.

When you change any of the INTERVAL values (and also when CICS is initialized), the length of the current (or first) interval is adjusted so that it expires after an integral number of intervals from the end-of-day time.

These rules are illustrated by the following example. *I* indicates an interval recording and *E* indicates an end-of-day recording.

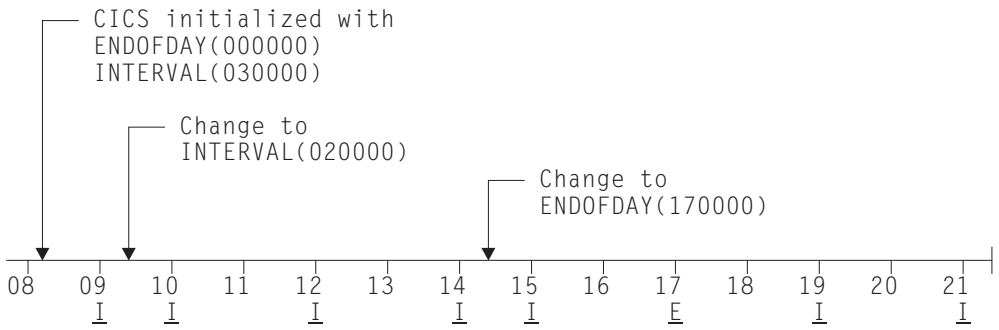


Figure 2. Resetting statistics counters

If you want your end-of-day recordings to cover 24 hours, set INTERVAL to 240000.

Note: Interval statistics are taken precisely on a minute boundary. Thus users with many CICS regions on a single MVS image could have every region writing statistics at the same time, if you have both the same interval and the same end of day period specified. This could cost up to several seconds of the entire CPU. If the cost becomes too noticeable, in terms of user response time around the interval expiry, you should consider staggering the intervals. One way of doing this while still maintaining very close correlation of intervals for all regions is to use a PLT program like the supplied sample DFH$STED which changes the end-of-day, and thus each interval expiry boundary, by a few seconds. See the *CICS Operations and Utilities Guide* for further information about DFH$STED.

Setting STATRCD=OFF reduces the number of times that statistics are written to the SMF data set and the counters are reset, to the end-of-day only.

Processing CICS statistics

There are four ways of processing CICS statistics:

1. Use the CICS DFHSTUP offline utility. For guidance about retrieving CICS statistics from SMF, and about running DFHSTUP, see the *CICS Operations and Utilities Guide*.
2. Write your own program to report and analyze the statistics. For details about the statistics record types, see the assembler DSECTs named in each set of statistics. For programming information about the formats of CICS statistics SMF records, see the *CICS Customization Guide*.
3. Use the sample statistics program (DFH0STAT).

   You can use the statistics sample program, DFH0STAT, to help you determine and adjust the values needed for CICS storage parameters, for example, using DSALIM and EDSALIM. The program produces a report showing critical system parameters from the CICS dispatcher, an analysis of the CICS storage manager and loader statistics, and an overview of the MVS storage in use. The program demonstrates the use of the EXEC CICS INQUIRE and EXEC CICS COLLECT STATISTICS commands to produce an analysis of a CICS system. You can use the sample program as provided or modify it to suit your needs. In addition, the DFH0STAT report produces reports from the CICS statistics reports. For more information about these, see "Appendix E. The sample statistics program, DFH0STAT" on page 513.
4. Use the Performance Reporter program to process CICS SMF records to produce joint reports with data from other SMF records. For more information, see "Chapter 7. Tivoli Performance Reporter for OS/390" on page 109.

Interpreting CICS statistics

In the following sections, as indicated in Table 1, guidance is given to help with the interpretation of the statistics report. Information is presented in the order that it appears in the DFHSTUP report. Some headings have been omitted where they have little or no performance impact. Detailed information about the statistics tables is given in "Appendix A. CICS statistics tables" on page 339.

Table 1. Performance statistics types

Statistic type	page
CICS DB2 statistics	43
Dispatcher statistics	43
Dump statistics	48
Enqueue domain statistics	48
Front end programming interface statistics	49
Files	49
ISC/IRC attach time statistics	59
Journalname and logstream statistics	50
Loader statistics	44
LSRPOOLS	51
Programs	49

Table 1. Performance statistics types (continued)

Statistic type	page
Recovery manager statistics	52
Shared TS queue server statistics	59
Statistics domain statistics	42
Storage manager statistics	44
Temporary storage	45
Terminals	52
Transaction class statistics	42
Transaction manager statistics	42
Transactions	49
Transient data (global)	46
Transient data (resource)	46
User domain statistics	46
VTAM statistics	47

Statistics domain statistics

Statistics recording on to an SMF data set can be a very CPU-intensive activity. The amount of activity depends more on the number of resources defined than the extent of their use. This may be another reason to maintain CICS definitions by removing redundant or over-allocated resources.

For more information about the statistics domain statistics, see page 445.

Transaction manager statistics

The “Times the MAXTASK limit reached” indicates whether MXT is constraining your system, or any possible integrity exposures are resulting from forced resolutions of UOWs relating to the transactions. The only time that you may need to constrain your system in this way is to reduce virtual storage usage. As most CICS virtual storage is above the 16MB line you may be able to run your system without MXT constraints, but note that CICS does preallocate storage, above and below the 16MB line, for each MXT whether or not it is used. Changing MXT affects your calculations for the dynamic storage areas. See “Maximum task specification (MXT)” on page 285 for more information.

For more information about transaction manager statistics, see page 476.

Transaction class (TRANCLASS) statistics

If you are never at the limit of your transaction class setting then you might consider resetting its value, or review whether there is any need to continue specifying any transaction types with that class.

For more information, see the transaction class statistics on page 472

CICS DB2 statistics

In addition to the limited statistics output by the DSNC DISP STAT command and those output to the STATSQUEUE destination of the DB2CONN during attachment facility shutdown, a more comprehensive set of CICS DB2 statistics can be collected using standard CICS statistics interfaces:

- The EXEC CICS COLLECT statistics command accepts the DB2CONN keyword to allow CICS DB2 global statistics to be collected. CICS DB2 global statistics are mapped by the DFHD2GDS DSECT.
- The EXEC CICS COLLECT statistics command accepts the DB2ENTRY() keyword to allow CICS DB2 resource statistics to be collected for a particular DB2ENTRY. CICS DB2 resource statistics are mapped by the DFHD2RDS DSECT.
- The EXEC CICS PERFORM STATISTICS command accepts the DB2 keyword to allow the user to request that CICS DB2 global and resource statistics are written out to SMF.

The CICS DB2 global and resource statistics are described in the CICS statistics tables on page 346. For more information about CICS DB2 performance, see the *CICS DB2 Guide*.

Dispatcher statistics

TCB statistics

The "Accum CPU time/TCB" is the amount of CPU time consumed by each CICS TCB since the last time statistics were reset. Totaling the values of "Accum time in MVS wait" and "Accum time dispatched" gives you the approximate time since the last time CICS statistics were reset. The ratio of the "Accum CPU time /TCB" to this time shows the percentage usage of each CICS TCB. The "Accum CPU time/TCB" does not include uncaptured time, thus even a totally busy CICS TCB would be noticeably less than 100% busy from this calculation. If a CICS region is more than 70% busy by this method, you are approaching that region's capacity. The 70% calculation can only be very approximate, however, depending on such factors as the workload in operation, the mix of activity within the workload, and which release of CICS you are currently using. Alternatively, you can calculate if your system is approaching capacity by using RMF to obtain a definititve measurement, or you can use RMF with your monitoring system. For more information, see *OS/390 RMF V2R6 Performance Management Guide*, *SC28-1951*.

Note: "Accum time dispatched" is NOT a measurement of CPU time because MVS can run higher priority work, for example, all I/O activity and higher priority regions, without CICS being aware.

Modes of TCB are as follows:

QR mode
There is always one quasi-reentrant mode TCB. It is used to run quasi-reentrant CICS code and non-threadsafe application code.

FO mode
There is always one file-owning TCB. It is used for opening and closing user datasets.

RO mode
There is always one resource-owning TCB. It is used for opening and closing CICS datasets, loading programs, issuing RACF® calls, etc.

CO mode
The optional concurrent mode TCB is used for processes which can safely run in parallel with other CICS activity such as VSAM requests. The SIT keyword SUBTSKS has been defined to have numeric values (0 and 1) to specify whether there is to be a CO TCB.

SZ mode
The single optional SZ mode TCB is used by the FEPI interface.

RP mode
The single optional RP mode TCB is used to make ONC/RPC calls.

J8 mode
A task has a J8 mode TCB for its sole use if it needs to run a JVM.

L8 mode
L8 mode TCBs are not in use for CICS Transaction Server for OS/390 Release 3.

SO mode
The SO mode TCB is used to make calls to the sockets interface of TCP/IP.

SL mode
The SL mode TCB is used to wait for activity on a set of listening sockets.

S8 mode
A task has an S8 TCB for its sole use if it needs to use the system Secure Sockets Layer..

For more information about dispatcher statistics, see page 360.

Storage manager statistics

Dynamic program compression releases programs which are not being used progressively as storage becomes shorter. However, short-on-storage conditions can still occur and are reported as "Times went short on storage". If this value is not zero you might consider increasing the size of the dynamic storage area. Otherwise you should consider the use of MXT and transaction classes to constrain your system's virtual storage.

Storage manager requests "Times request suspended", and "Times cushion released", indicate that storage stress situations have occurred, some of which may not have produced a short-on-storage condition. For example, a GETMAIN request may cause the storage cushion to be released. However, loader can compress some programs, obtain the cushion storage, and avoid the short-on-storage condition.

Note: In the task subpools section, the "Current elem stg" is the number of bytes actually used while "Current page stg" is the number of pages containing one or more of these bytes.

For more information, see the CICS statistics tables on page 446.

Loader statistics

"Average loading time" = "Total loading time" / "Number of library load requests". This indicates the response time overhead suffered by tasks when accessing a program which has to be brought into storage. If "Average loading time" has increased over a period, consider MVS library lookaside usage. "Not-in-use" program storage is freed progressively so that the "Amount of the

dynamic storage area occupied by not in use programs", and the free storage in the dynamic storage area are optimized for performance. Loader attempts to keep not-in-use programs in storage long enough to reduce the performance overhead of reloading the program. As the amount of free storage in the dynamic storage decreases, the not-in-use programs are freemained in order of those least frequently used to avoid a potential short-on-storage condition.

Note: The values reported are for the instant at which the statistics are gathered and vary since the last report.

"Average Not-In-Use queue membership time" = "Total Not-In-Use queue membership time" / "Number of programs removed by compression". This is an indication of how long a program is left in storage when not in use before being removed by the dynamic program storage compression (DPSC) mechanism. If the interval between uses of a program, that is, interval time divided by the number of times used in the interval, is less than this value, there is a high probability that the program is in storage already when it is next required.

Note: This factor is meaningful only if there has been a substantial degree of loader domain activity during the interval and may be distorted by startup usage patterns.

"Average suspend time" = "Total waiting time" / "Number of waited loader requests".

This is an indication of the response time impact which may be suffered by a task due to contention for loader domain resources.

Note: This calculation is not performed on requests that are currently waiting.

For more information, see the CICS statistics tables on page 424.

Temporary storage statistics

If a data item is written to temporary storage (using WRITEQ TS), a temporary storage queue is built.

The "Writes more than control interval" is the number of writes of records whose length was greater than the control interval (CI) size of the TS data set. This value should be used to adjust the CI size. If the reported value is large, increase the CI size. If the value is zero, consider reducing the CI size until a small value is reported.

The number of "times aux. storage exhausted" is the number of situations where one or more transactions may have been suspended because of a NOSPACE condition, or (using a HANDLE CONDITION NOSPACE command, the use of RESP on the WRITEQ TS command, or WRITEQ TS NOSUSPEND command) may have been forced to abend. If this item appears in the statistics, increase the size of the temporary storage data set. "Buffer writes" is the number of WRITEs to the temporary storage data set. This includes both WRITEs necessitated by recovery requirements and WRITEs forced by the buffer being needed to accommodate another CI. I/O activity caused by the latter reason can be minimized by increasing buffer allocation using the system initialization parameter, TS=(b,s), where b is the number of buffers and s is the number of strings.

The "Peak number of strings in use" item is the peak number of concurrent I/O operations to the data set. If this is significantly less than the number of strings specified in the TS system initialization parameter, consider reducing the system initialization parameter to approach this number.

If the "Times string wait occurred" is not zero, consider increasing the number of strings. For details about adjusting the size of the TS data set and the number of strings and buffers, see the *CICS System Definition Guide*.

For more information, see the CICS statistics tables on page 462

Transient data statistics

You should monitor the data provided by CICS on the amount of I/O activity for transient data, in the form of the number of READs and WRITEs to the transient data intrapartition data set. If there is a large amount of READ activity, this indicates that the buffer allocation may be insufficient, even though the "peak concurrent string access" may be fewer than the number allocated.

You should aim to minimize the "Intrapartition buffer waits" and "string waits" by increasing the number of buffers and the number of strings if you can afford any associated increase in your use of real storage.

For more information, see the CICS statistics tables on pages 497 and 462.

User domain statistics

The user domain attempts to minimize the number of times it calls the security domain to create user security blocks (such as the ACEE), because this operation is very expensive in both processor time and input/output operations. If possible, each unique representation of a user is shared between multiple transactions. A user-domain representation of a user can be shared if the following attributes are identical:

- The userid.
- The groupid.
- The applid. This is not necessarily the same for all the users in a region. The applid is shipped with the userid across MRO links.
- The port of entry. This can be the netname for users signed on at VTAM terminals, or the console name for users signed on at consoles. It is null for other terminal types and for users associated with non-terminal transactions.

The user domain keeps a count of the number of concurrent usages of a shared instance of a user. The count includes the number of times the instance has been associated with a CICS resource (such as a transient data queue) and the number of active transactions that are using the instance.

Whenever CICS adds a new user instance to the user domain, the domain attempts to locate that instance in its user directory. If the user instance already exists with the parameters described above, that instance is reused. USGDRRC records how many times this is done. However, if the user instance does not already exist, it needs to be added. This requires an invocation of the security domain and the external security manager. USGDRNFC records how many times this is necessary.

When the count associated with the instance is reduced to zero, the user instance is not immediately deleted: instead it is placed in a timeout queue controlled by the

USRDELAY system initialization parameter. While it is in the timeout queue, the user instance is still eligible to be reused. If it is reused, it is removed from the timeout queue. USGTORC records how many times a user instance is reused while it was being timed out, and USGTOMRT records the average time that user instances remain on the timeout queue until they are removed.

However, if a user instance remains on the timeout queue for a full USRDELAY interval without being reused, it is deleted. USGTOEC records how many times this happens.

If USGTOEC is large compared to USGTORC, you should consider increasing the value of USRDELAY. But if USGTOMRT is much smaller than USRDELAY, you may be able to reduce USRDELAY without significant performance effect.

You should be aware that high values of USRDELAY may affect your security administrator's ability to change the authorities and attributes of CICS users, because those changes are not reflected in CICS until the user instance is refreshed in CICS by being flushed from the timeout queue after the USRDELAY interval. Some security administrators may require you to specify USRDELAY=0. This still allows some sharing of user instances if the usage count is never reduced to zero. Generally, however, remote users are flushed out immediately after the transaction they are executing has terminated, so that their user control blocks have to be reconstructed frequently. This results in poor performance. For more information, see "User domain statistics" on page 493.

VTAM statistics

The "peak RPLs posted" includes only the receive-any RPLs defined by the RAPOOL system initialization parameter. In non-HPO systems, the value shown can be larger than the value specified for RAPOOL, because CICS reissues each receive-any request as soon as the input message associated with the posted RPL has been disposed of. VTAM may well cause this reissued receive-any RPL to be posted during the current dispatch of terminal control. While this does not necessarily indicate a performance problem, a number much higher than the number of receive-any requests specified via RAPOOL may indicate, for MVS, that VTAM was required to queue incoming messages in subpool 229 when no receive-any was available to accept the input. You should limit this VTAM queueing activity by providing a sufficient number of receive-any requests to handle all but the input message rate peaks.

In addition to indicating whether the value for the RAPOOL system initialization parameter is large enough, you can also use the "maximum number of RPLs posted" statistic (A03RPLX) to determine other information. This depends upon whether your MVS system has HPO or not.

For HPO, RAPOOL(A,B) allows the user to tune the active count (B). The size of the pool (A) should be dependent on the speed at which they get processed. The active count (B) has to be able to satisfy VTAM at any given time, and is dependent on the inbound message rate for receive-any requests.

Here is an example to illustrate the differences for an HPO and a non-HPO system. Suppose two similar CICS executions use a RAPOOL value of 2 for both runs. The number of RPLs posted in the MVS/HPO run is 2, while the MVS/non-HPO run is 31. This difference is better understood when we look at the next item in the statistics.

This item is not printed if the maximum number of RPLs posted is zero. In our example, let us say that the MVS/HPO system reached the maximum 495 times. The non-HPO MVS system reached the maximum of 31 only once. You might deduce from this that the pool is probably too small (RAPOOL=2) for the HPO system and it needs to be increased. An appreciable increase in the RAPOOL value, from 2 to, say, 6 or more, should be tried. As you can see from the example given below, the RAPOOL value was increased to 8 and the maximum was reached only 16 times:

```
MAXIMUM NUMBER OF RPLS POSTED         8
NUMBER OF TIMES REACHED MAXIMUM      16
```

In a non-HPO system, these two statistics are less useful, except that, if the maximum number of RPLs posted is less than RAPOOL, RAPOOL can be reduced, thereby saving virtual storage.

VTAM SOS simply means that a CICS request for service from VTAM was rejected with a VTAM sense code indicating that VTAM was unable to acquire the storage required to service the request. VTAM does not give any further information to CICS, such as what storage it was unable to acquire.

This situation most commonly arises at network startup or shutdown when CICS is trying to schedule requests concurrently, to a larger number of terminals than during normal execution. If the count is not very high, it is probably not worth tracking down. In any case, CICS automatically retries the failing requests later on.

If your network is growing, however, you should monitor this statistic and, if the count is starting to increase, you should take action. Use D NET,BFRUSE to check if VTAM is short on storage in its own region and increase VTAM allocations accordingly if this is required.

The maximum value for this statistic is 99, at which time a message is sent to the console and the counter is reset to zero. However, VTAM controls its own buffers and gives you a facility to monitor buffer usage.

If you feel that D NET,BFRUSE is insufficient, you can activate SMS tracing in VTAM to sample buffer activity at regular intervals. If you have installed NetView, you can also have dynamic displays of the data that is obtained with D NET, BFRUSE.

For more information, see the CICS statistics tables on page 494.

Dump statistics

Both transaction and system dumps are very expensive and should be thoroughly investigated and eliminated.

For more information, see the CICS statistics tables on page 367.

Enqueue statistics

The enqueue domain supports the CICS recovery manager. Enqueue statistics contain the global data collected by the enqueue domain for enqueue requests.

Waiting for an enqueue on a resource can add significant delays in the execution of a transaction. The enqueue statistics allow you to assess the impact of waiting for

enqueues in the system and the impact of retained enqueues on waiters. Both the current activity and the activity since the last reset are available.

For more information, see the CICS statistics tables on page 372.

Transaction statistics

Use these statistics to find out which transactions (if any) had storage violations.

It is also possible to use these statistics for capacity planning purposes. But remember, many systems experience both increasing cost per transaction as well as increasing transaction rate.

For more information, see the CICS statistics tables on page 478.

Program statistics

"Average fetch time" is an indication of how long it actually takes MVS to perform a load from the partitioned data set in the RPL concatenation into CICS managed storage.

The average for each RPL offset of "Program size" / "Average fetch time". is an indication of the byte transfer rate during loads from a particular partitioned data set. A comparison of these values may assist you to detect bad channel loading or file layout problems.

For more information, see the CICS statistics tables on page 436.

Front end programming interface (FEPI) statistics

CICS monitoring and statistics data can be used to help tune FEPI applications, and to control the resources that they use. FEPI statistics contain data about the use of each FEPI pool, a particular target in a pool, and each FEPI connection.

For more information, see the CICS statistics tables on page 376.

File statistics

File statistics collect data about the number of application requests against your data sets. They indicate the number of requests for each type of service that are processed against each file. If the number of requests is totalled daily or for every CICS execution, the activity for each file can be monitored for any changes that occur. Note that these file statistics may have been reset during the day; to obtain a figure of total activity against a particular file during the day, refer to the DFHSTUP summary report. Other data pertaining to file statistics and special processing conditions are also collected.

The wait-on-string number is only significant for files related to VSAM data sets. For VSAM, STRNO=5 in the file definition means, for example, that CICS permits five concurrent requests to this file. If a transaction issues a sixth request for the same file, this request must wait until one of the other five requests has completed ("wait-on-string").

The number of strings associated with a file when specified through resource definition online.

String number setting is important for performance. Too low a value causes excessive waiting for strings by tasks and long response times. Too high a value increases VSAM virtual storage requirements and therefore real storage usage. However, as both virtual storage and real storage are above the 16MB line, this may not be a problem. In general, the number of strings should be chosen to give near zero "wait on string" count.

Note: Increasing the number of strings can increase the risk of deadlocks because of greater transaction concurrency. To minimize the risk you should ensure that applications follow the standards set in the *CICS Application Programming Guide*.

A file can also "wait-on-string" for an LSRpool string. This type of wait is reflected in the local shared resource pool statistics section (see "LSRPOOL statistics" on page 51) and not in the file wait-on-string statistics.

If you are using data tables, an extra line appears in the DFHSTUP report for those files defined as data tables. "Read requests", "Source reads", and "Storage alloc(K)" are usually the numbers of most significance. For a CICS-maintained table a comparison of the difference between "read requests" and "source reads" with the total request activity reported in the preceding line shows how the request traffic divides between using the table and using VSAM and thus indicates the effectiveness of converting the file to a CMT. "Storage alloc(K)" is the total storage allocated for the table and provides guidance to the cost of the table in storage resource, bearing in mind the possibility of reducing LSRpool sizes in the light of reduced VSAM accesses.

For more information, see the CICS statistics tables on page 379.

Journalname and log stream statistics

CICS collects statistics on the data written to each journal and log stream which can be used to analyze the activity of a single region. However, because log streams can be shared across multiple MVS images, it can be more useful to examine the statistics generated by MVS.

Journalname statistics contain data about the use of each journal, as follows:

- The journal type (MVS logger, SMF or dummy)
- The log stream name for MVS logger journal types only
- The number of API journal writes
- The number of bytes written
- The number of flushes of journal data to log streams or SMF.

Note that the CICS system journalname and log stream statistics for the last three items on this list are always zero. These entries appear in journalname statistics to inform you of the journal type and log stream name for the special CICS system journals.

For more information on journalname statistics, see the CICS statistics tables on page 405.

Log stream statistics contain data about the use of each log stream including the following:

- The number of write requests to the log stream

- The number of bytes written to the log stream
- The number of log stream buffer waits
- The number of log stream browse and delete requests.

For more information on log stream statistics, see the CICS statistics tables on page 407.

Journalnames are a convenient means of identifying a destination log stream that is to be written to. CICS applications write data to journals using their journalname. CICS itself usually uses the underlying log stream name when issuing requests to the CICS log manager, and this must be considered when interpreting journalname and log stream resource statistics. For example, these may show many operations against a log stream, but relatively few, if any, writes to a journalname which maps to that log stream. This indicates that it is CICS that accesses the resource at the log stream level, not an application writing to it through the CICS application programming interface. These results can typically be seen when examining the journalname resource statistics for DFHLOG and DFHSHUNT, and comparing them with the resource statistics for their associated CICS system log streams.

For more information on logging and journaling, see "Chapter 23. Logging and journaling" on page 269.

For information about the SMF Type 88 records produced by the MVS system logger, see the *OS/390 MVS System Management Facilities (SMF)* manual.

LSRPOOL statistics

CICS supports the use of up to eight LSRpools. CICS produces two sets of statistics for LSRpool activity: one set detailing the activity for each LSRpool, and one set giving details for each file associated with an LSRpool. Statistics are printed for all pools that have been built (a pool is built when at least one file using the pool has been opened).

You should usually aim to have no requests that waited for a string. If you do then the use of MXT may be more effective.

When the last open file in an LSRPOOL is closed, the pool is deleted. The subsequent unsolicited statistics (USS) LSRPOOL record written to SMF can be mapped by the DFHA08DS DSECT.

The fields relating to the size and characteristics of the pool (maximum key length, number of strings, number and size of buffers) may be those which you have specified for the pool, through resource definition online command DEFINE LSRPOOL. Alternatively, if some, or all, of the fields were not specified, the values of the unspecified fields are those calculated by CICS when the pool is built.

It is possible to change the LSRPOOL specification of a file when it is closed, but you must then consider the characteristics of the pool that the file is to share if the pool is already built, or the file open may fail. If the pool is not built and the pool characteristics are specified by you, take care that these are adequate for the file. If the pool is not built and CICS calculates all or some of the operands, it may build the pool creations of that pool. The statistics show all creations of the pool, so any changed characteristics are visible.

You should consider specifying separate data and index buffers if you have not already done so. This is especially true if index CI sizes are the same as data CI sizes.

You should also consider using Hiperspace™ buffers while retaining a reasonable number of address space buffers. Hiperspace buffers tend to give CPU savings of keeping data in memory, exploiting the relatively cheap expanded storage, while allowing central storage to be used more effectively.

For more information, see the CICS statistics tables on pages 410.

Recovery manager statistics

Recovery manager statistics detail the syncpoint activity of all the transactions in the system. From these statistics you can assess the impact of shunted UOWs (units of work that suffered an indoubt failure and are waiting for resynchronization with their recovery coordinator, or for the problem with the resources to be resolved). Shunted UOWs still hold locks and enqueues until they are resolved. Statistics are available on any forced resolutions of shunted UOWs to help assess whether any integrity exposures may have been introduced. The current activity and the activity since the last reset are available.

For more information, see the CICS statistics tables on page 439.

Terminal statistics

There are a number of ways in which terminal statistics are important for performance analysis. From them, you can get the number of inputs and outputs, that is, the loading of the system by end users. Line-transmission faults and transaction faults are shown (these both have a negative influence on performance behavior).

For more information, see the CICS statistics tables on page 468.

ISC/IRC system and mode entry statistics

You can use the ISC/IRC system and mode entry statistics to detect some problems in a CICS intersystem environment.

The following section attempts to identify the kind of questions you may have in connection with system performance, and describes how answers to those questions can be derived from the statistics report. It also describes what actions, if any, you can take to resolve ISC/IRC performance problems.

Some of the questions you may be seeking an answer to when looking at these statistics are these:

- Are there enough sessions defined?
- Is the balance of *contention winners* to *contention losers* correct?
- Is there conflicting usage of APPC modegroups?
- What can be done if there are unusually high numbers, compared with normal or expected numbers, in the statistics report?

Summary connection type for statistics fields

The following two tables show the connection type that is relevant for each statistics field:

Table 2. ISC/IRC system entries

System entry	Field	IRC	LU6.1	APPC
Connection name	A14CNTN	X	X	X
AIDS in chain	A14EALL	X	X	X
Generic AIDS in chain	A14ESALL	X	X	X
ATIs satisfied by contention losers	A14ES1		X	
ATIs satisfied by contention winners	A14ES2	X	X	
Peak contention losers	A14E1HWM	X	X	
Peak contention winners	A14E2HWM	X	X	
Peak outstanding allocates	A14ESTAM	X	X	X
Total number of allocates	A14ESTAS	X	X	X
Queued allocates	A14ESTAQ	X	X	X
Failed link allocates	A14ESTAF	X	X	X
Failed allocates due to sessions in use	A14ESTAO	X	X	X
Total bids sent	A14ESBID		X	
Current bids in progress	A14EBID		X	
Peak bids in progress	A14EBHWM		X	
File control function shipping requests	A14ESTFC	X	X	X
Interval control function shipping requests	A14ESTIC	X	X	X
TD function shipping requests	A14ESTTD	X	X	X
TS function shipping requests	A14ESTTS	X	X	X
DLI function shipping requests	A14ESTDL	X	X	X
Terminal sharing requests	A14ESTTC	X		X

All the fields below are specific to the mode group of the mode name given.

Table 3. ISC/IRC mode entries

Mode entry	Field	IRC	LU6.1	APPC
Mode name	A20MODE			X
ATIs satisfied by contention losers	A20ES1			X
ATIs satisfied by contention winners	A20ES2			X
Peak contention losers	A20E1HWM			X
Peak contention winners	A20E2HWM			X
Peak outstanding allocates	A20ESTAM			X
Total specific allocate requests	A20ESTAS			X
Total specific allocates satisfied	A20ESTAP			X
Total generic allocates satisfied	A20ESTAG			X
Queued allocates	A20ESTAQ			X

Table 3. ISC/IRC mode entries (continued)

Mode entry	Field	IRC	LU6.1	APPC
Failed link allocates	A20ESTAF			X
Failed allocates due to sessions in use	A20ESTAO			X
Total bids sent	A20ESBID			X
Current bids in progress	A20EBID			X
Peak bids in progress	A20EBHWM			X

For more information about the usage of individual fields, see the CICS statistics described under "ISC/IRC system and mode entries" on page 390.

General guidance for interpreting ISC/IRC statistics

Here is some guidance information on interpreting the ISC/IRC statistics:

1. Usage of A14xxx and A20xxx fields:
   - In most cases, the guidance given in the following section relates to all connection types, that is, IRC, LU6.1, and APPC. Where the guidance is different for a particular connection type, the text indicates the relevant type of connection.
   - The statistics fields that relate to IRC and LU6.1 are always prefixed A14, whereas the APPC fields can be prefixed by A14 or A20. For more information on which field relates to which connection type, see Table 2 on page 53 and Table 3 on page 53.
2. Use of the terms "Contention Winner" and "Contention Loser":
   - APPC sessions are referred to as either *contention winners* or *contention losers*. These are equivalent to secondaries (SEND sessions) and primaries (RECEIVE sessions) when referring to LU6.1 and IRC.
3. Tuning the number of sessions defined:
   - In the following sections, it is sometimes stated that, if certain counts are too high, you should consider making more sessions available. In these cases, be aware that, as the number of sessions defined in the system is increased, it may have the following effects:
     - Increased use of real and virtual storage.
     - Increased use of storage on GATEWAY NCPs in the network.
     - Increased use of storage by VTAM.
     - Increased line loading in the network.
     - The back-end CICS system (AOR) may not be able to cope with the increased workload from the TOR.
     - Possible performance degradation due to increased control block scanning by CICS.
   - The recommendation is to set the number of sessions available to the highest value you think you may need and then, through monitoring the statistics (both ISC/IRC and terminal statistics) over a number of CICS runs, reduce the number of sessions available to just above the number required to avoid problems.
4. Tuning the number of contention winner and contention loser sessions available:
   - Look at both sides of the connection when carrying out any tuning, because changing the loading on one side could inversely affect the other. Any change made to the number of contention winner sessions available in the TOR has an effect on the number of contention loser sessions in the AOR.

5. Establish a connection profile for comparison and measurement.

   One of the objectives of a tuning exercise should be to establish a profile of the usage of CICS connections during both normal and peak periods. Such usage profiles can then be used as a reference point when analyzing statistics to help you:

   - Determine changed usage patterns over a period of time
   - Anticipate potential performance problems before they become critical.

Are enough sessions defined?

To help you determine whether you have enough sessions defined, you can check a number of peak fields that CICS provides in the statistics report. These are:

1. *"Peak outstanding allocates"* (fields A14ESTAM and A20ESTAM) *"Total number of allocates"* (field A14ESTAS) *"Total specific allocate requests"* (field A20ESTAS).

   When reviewing the number of sessions for APPC modegroups, and the number of "Peak outstanding allocates" appears high in relation to the "Total number of allocates", or the "Total specific allocate requests" within a statistics reporting period, it could indicate that the total number of sessions defined is too low.

2. *"Peak contention winners"* (fields A14E2HWM and A20E2HWM) *"Peak contention losers"* (fields A14E1HWM and A20E1HWM)

   If the number of ("Peak contention winners" + "Peak contention losers") equals the maximum number of sessions available (as defined in the SESSIONS definition), this indicates that, at some point in the statistics reporting period, all the sessions available were, potentially, in use. While these facts alone may not indicate a problem, if CICS also queued or rejected some allocate requests during the same period, the total number of sessions defined is too low.

3. *"Failed allocates due to sessions in use"* (fields A14ESTAO and A20ESTAO)

   This value is incremented for allocates that are rejected with a SYSBUSY response because no sessions are immediately available (that is, for allocate requests with the NOSUSPEND or NOQUEUE option specified). This value is also incremented for allocates that are queued and then rejected with an AAL1 abend code; the AAL1 code indicates the allocate is rejected because no session became available within the specified deadlock timeout (DTIMOUT) time limit.

   If the number of "Failed allocates due to sessions in use" is high within a statistics reporting period, it indicates that not enough sessions were immediately available, or available within a reasonable time limit.

Action: Consider making more sessions available with which to satisfy the allocate requests. Enabling CICS to satisfy allocate requests without the need for queueing may lead to improved performance.

However, be aware that increasing the number of sessions available on the front end potentially increases the workload to the back end, and you should investigate whether this is likely to cause a problem.

Is the balance of contention winners to contention losers correct?

There are several ways to determine the answer to this, because CICS provides a number of fields which show contention winner and contention loser usage.

The following fields should give some guidance as to whether you need to increase the number of contention winner sessions defined:

1. *"Current bids in progress"* (fields A14EBID and A20EBID) *"Peak bids in progress"* (fields A14EBHWM and A20EBHWM)

   The value "Peak bids in progress" records the maximum number of bids in progress at any one time during the statistics reporting period. "Current bids in progress" is always less than or equal to the "Peak bids in progress".

   Ideally, these fields should be kept to zero. If either of these fields is high, it indicates that CICS is having to perform a large number of bids for contention loser sessions.

2. *"Peak contention losers"* (fields A14E1HWM and A20E1HWM).

   If the number of "Peak contention losers" is equal to the number of contention loser sessions available, the number of contention loser sessions defined may be too low. Alternatively, for APPC/LU6.1, CICS could be using the contention loser sessions to satisfy allocates due to a lack of contention winner sessions. This should be tuned at the front-end in conjunction with winners at the back-end. For details of how to specify the maximum number of sessions, and the number of contention winners, see the information on defining SESSIONS in the *CICS Resource Definition Guide*.

Actions:

For APPC, consider making more contention winner sessions available, which should reduce the need to use contention loser sessions to satisfy allocate requests and, as a result, should also make more contention loser sessions available.

For LU6.1, consider making more SEND sessions available, which decreases the need for LU6.1 to use primaries (RECEIVE sessions) to satisfy allocate requests.

For IRC, there is no bidding involved, as MRO can never use RECEIVE sessions to satisfy allocate requests. If "Peak contention losers (RECEIVE)" is equal to the number of contention loser (RECEIVE) sessions on an IRC link, the number of allocates from the remote system is possibly higher than the receiving system can cope with. In this situation, consider increasing the number of RECEIVE sessions available.

Note: The usage of sessions depends on the direction of flow of work. Any tuning which increases the number of winners available at the front-end should also take into account whether this is appropriate for the direction of flow of work over a whole period, such as a day, week, or month.

Is there conflicting usage of APPC modegroups?

There is a possibility of conflicting APPC modegroup usage, where a mixture of generic and specific allocate requests is used within a CICS region.

A specific allocate is an allocate request that specifies a particular (specific) mode group of sessions to allocate from, whereas a generic allocate does not specify any particular mode group only the system to which an allocate is required. In the latter case CICS determines the session and mode group to allocate.

The fields you need to investigate to answer this question, are:

"Total generic allocates satisfied" (field A20ESTAG)

"Total specific allocate requests" (field A20ESTAS)

"Peak outstanding allocates" (field A20ESTAM)

"Total specific allocates satisfied" (field A20ESTAP).

If the "Total generic allocates satisfied" is much greater than "Total specific allocate requests", and "Peak outstanding allocates" is not zero, it could indicate that generic allocates are being made only, or mainly, to the first modegroup for a connection.

This could cause a problem for any specific allocate, because CICS initially tries to satisfy a generic allocate from the first modegroup before trying other modegroups in sequence.

Action: Consider changing the order of the installed modegroup entries. Modegroups for a connection are represented by TCT mode entries (TCTMEs), with the modegroup name being taken from the MODENAME specified on the SESSIONS definition. The order of the TCTMEs is determined by the order in which CICS installs the SESSIONS definitions, which is in the order of the SESSIONS name as stored on the CSD (ascending alphanumeric key sequence). See Figure 3 for an illustration of this. To change the order of the TCTMEs, you must change the names of the SESSIONS definitions. You can use the CEDA RENAME command with the AS option to rename the definition with a different SESSIONS name within the CSD group. By managing the order in which the TCTMEs are created you can ensure that specific allocates reference modegroups lower down the TCTME chain, and avoid conflict with the generic ALLOCATEs. *Alternatively, make all allocates specific allocates*.

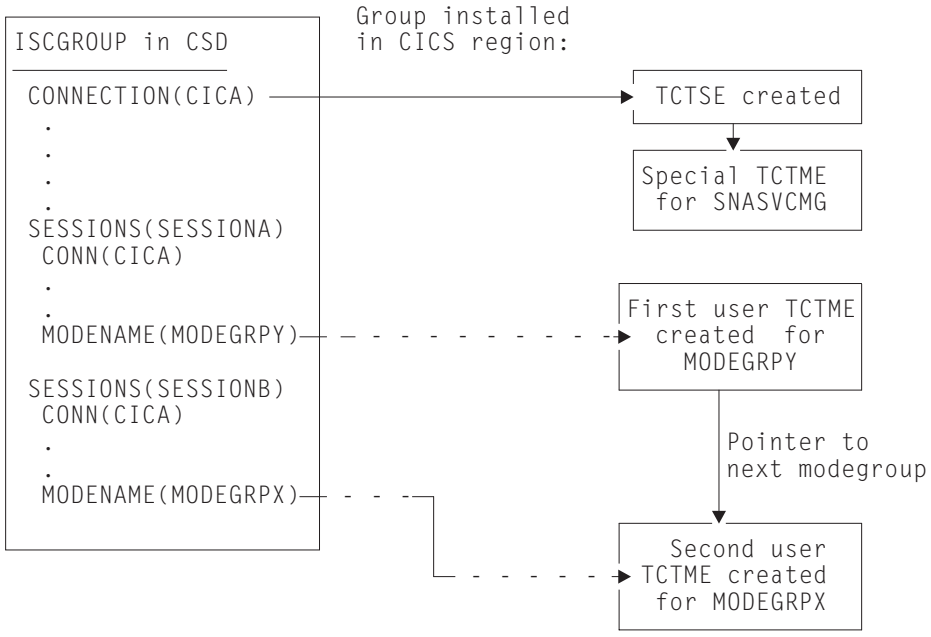


Figure 3. How the sequence of TCT mode entries is determined

What if there are unusually high numbers in the statistics report?

When looking down the *ISC/IRC system and mode entries* statistics report, you may notice a number of fields that appear to be unusually high in relation to all others. This section lists some of those fields, and what action you can take to reduce their numbers:

1. *"Peak contention losers"* (fields A14E1HWM and A20E1HWM).

   If the number of "Peak contention losers" is equal to the number of contention loser sessions available, the number of contention loser sessions defined may be

too low, or, if your links are APPC/LU6.1, CICS could be using the contention loser sessions to satisfy allocates due to a lack of contention winner sessions.

Action: Consider making more contention winner sessions available with which to satisfy the allocate requests. If IRC, increase the RECEIVES.

2. *"Peak outstanding allocates"* (fields A14ESTAM and A20ESTAM)

   If the number of "Peak outstanding allocates" appears high, in relation to the "Total number of allocates", or the "Total specific allocate requests" for APPC modegroups within a statistics reporting period, it could indicate that the total number of sessions defined is too low, or that the remote system cannot cope with the amount of work being sent to it.

   Action: Consider making more sessions available with which to satisfy the allocate requests, or reduce the number of allocates being made.

3. *"Failed link allocates"* (fields A14ESTAF and A20ESTAF)

   If this value is high within a statistics reporting period, it indicates something was wrong with the state of the connection. The most likely cause is that the connection is released, out of service, or has a closed mode group.

   Action: Examine the state of the connection that CICS is trying to allocate a session on, and resolve any problem that is causing the allocates to fail.

   To help you to resolve a connection failure, check the CSMT log for the same period covered by the statistics for any indication of problems with the connection that the statistics relate to.

   It may also be worth considering writing a connection status monitoring program, which can run in the background and regularly check connection status and take remedial action to re-acquire a released connection. This may help to minimize outage time caused by connections being unavailable for use. See the *CICS System Programming Reference* manual for programming information about the EXEC CICS INQUIRE|SET CONNECTION and the EXEC CICS INQUIRE|SET MODENAME commands that you would use in such a program.

4. *"Failed allocates due to sessions in use"* (fields A14ESTAO and A20ESTAO)

   This value is incremented for allocates that have been rejected with a SYSBUSY response because no sessions were immediately available, and the allocate requests were made with the NOSUSPEND or NOQUEUE option specified. This value is also incremented for allocates that have been queued and then rejected with an AAL1 abend code; the AAL1 code indicates the allocate was rejected because no session was available within the specified deadlock timeout (DTIMOUT) time limit.

   If the number of "Failed allocates due to sessions in use" is high, within a statistics reporting period, it indicates that not enough sessions were immediately available, or available within a reasonable time limit.

   Action: The action is to consider making more contention winner sessions available. This action would result in a reduction in the amount of bidding being carried out, and the subsequent usage of contention loser sessions. Increase the sessions if IRC is used.

5. *"Peak bids in progress"* (fields A14EBHWM and A20EBHWM)

   Ideally, these fields should be kept to zero. If either of these fields are high, it indicates that CICS is having to perform a large amount of bidding for sessions.

   Action: Consider making more contention winner sessions available, to satisfy allocate requests.

ISC/IRC attach time entries

ISC/IRC Signon activity. If the number of "entries reused" in signon activity is low, and the "entries timed out" value for signon activity is high, the value of the USRDELAY system initialization parameter should be increased. The "average reuse time between entries" gives some indication of the time that could be used for the USRDELAY system initialization parameter.

ISC Persistent verification (PV) activity. If the number of "entries reused" in the PV activity is low, and the "entries timed out" value is high, the PVDELAY system initialization parameter should be increased. The "average reuse time between entries" gives some indication of the time that could be used for the PVDELAY system initialization parameter.

Note: If there are a lot of either signed-on or PV-entries timed out, and not many reused, your performance may be degraded because of the need to make calls to an external security manager, such as RACF for security checking.

For more information, see the CICS statistics tables on page 404.

Shared temporary storage queue server statistics

Shared temporary storage queue server statistics are provided by the AXM page pool management routines for the pools AXMPGANY and AXMPGLOW. For more information, see "Appendix B. Shared temporary storage queue server statistics" on page 497.

Coupling facility data tables server statistics

Coupling facility data tables server statistics are provided by the AXM page pool management routines for the pools AXMPGANY and AXMPGLOW. For more information, see "Appendix C. Coupling facility data tables server statistics" on page 503.

Named counter sequence number server statistics

Named counter sequence number server statistics are provided by the AXM page pool management routines for the pools AXMPGANY and AXMPGLOW. For more information, see "Appendix D. Named counter sequence number server" on page 509.

Chapter 6. The CICS monitoring facility

This chapter is divided as follows:

Introduction to CICS monitoring

CICS monitoring collects data about the performance of all user- and CICS-supplied transactions during online processing for later offline analysis. The records produced by CICS monitoring are of the MVS System Management Facility (SMF) type 110, and are written to an SMF data set.

Note: Statistics records and some journaling records are also written to the SMF data set as type 110 records. You might find it particularly useful to process the statistics records and the monitoring records together, because statistics provide resource and system information that is complementary to the transaction data produced by CICS monitoring. The contents of the statistics fields, and the procedure for processing them, are described in "Appendix A. CICS statistics tables" on page 339.

Monitoring data is useful both for performance tuning and for charging your users for the resources they use.

The classes of monitoring data

Three types, or "classes", of monitoring data may be collected. These are *performance* class data, *exception* class data, and *SYSEVENT* data.

Performance class data

Performance class data is detailed transaction-level information, such as the processor and elapsed time for a transaction, or the time spent waiting for I/O. At least one performance record is written for each transaction that is being monitored.

Performance class data provides detailed, resource-level data that can be used for accounting, performance analysis, and capacity planning. This data contains information relating to individual task resource usage, and is completed for each task when the task terminates.

You can enable performance-class monitoring by coding MNPER=ON (together with MN=ON) in the system initialization table (SIT). Alternatively you can use either the (CEMT SET MONITOR ON PERF) or EXEC CICS SET MONITOR STATUS(ON) PERFCLASS(PERF) commands.

This information could be used periodically to calculate the charges applicable to different tasks. If you want to set up algorithms for charging users for resources used by them, you could use this class of data collection to update the charging information in your organization's accounting programs. (For previous versions of CICS, we did not recommend charging primarily on exact resource usage, because of the overheads involved in getting these figures.)

Exception class data

Exception class monitoring data is information on CICS resource shortages that are suffered by a transaction. This data highlights possible problems in CICS system operation and is intended to help you identify system constraints that affect the performance of your transactions. There is one exception record for each type of exception condition. The exception records are produced and written to SMF as soon as the resource shortage encountered by the transaction has been resolved. Exception records are produced for each of the following resource shortages:

- Wait for storage in the CDSA
- Wait for storage in the UDSA
- Wait for storage in the SDSA
- Wait for storage in the RDSA
- Wait for storage in the ECDSA
- Wait for storage in the EUDSA
- Wait for storage in the ESDSA
- Wait for storage in the ERDSA
- Wait for auxiliary temporary storage
- Wait for auxiliary temporary storage string
- Wait for auxiliary temporary storage buffer
- Wait for coupling facility data tables locking (request) slot
- Wait for coupling facility data tables non-locking (request) slot
- Wait for file buffer
- Wait for LSRPOOL string.
- Wait for file string

If the monitoring performance class is also being recorded, the performance class record for the transaction includes the total elapsed time the transaction was delayed by a CICS system resource shortage. This is measured by the exception class and the number of exceptions encountered by the transaction. The exception class records can be linked to the performance class records either by the transaction sequence number or by the network unit-of-work id. For more information on the exception class records, see "Exception class data" on page 102.

You can enable exception-class monitoring by coding the MNEXC=ON (together with MN=ON) system initialization parameters. Alternatively, you can use either the CEMT command. (CEMT SET MONITOR ON EXCEPT) or EXEC CICS SET MONITOR STATUS(ON) EXCEPTCLASS(EXCEPT).

The SYSEVENT class of monitoring data

SYSEVENT data is a special kind of transaction timing information.

CICS invokes the MVS System Resource Manager (SRM) macro SYSEVENT at the end of every transaction to record the elapsed time of the transaction.

You can enable SYSEVENT class monitoring by coding the MNEVE=ON (together with MN=ON) system initialization parameters. Alternatively, you can use either the CEMT command (CEMT SET MONITOR ON EVENT) or EXEC CICS SET MONITOR STATUS(ON) EVENTCLASS(EVENT).

If the SYSEVENT option is used, at the end of each transaction CICS issues a Type 55 (X'37') SYSEVENT macro. This records each transaction ID, the associated terminal ID, and the elapsed time duration of each transaction. This information is collected by the SRM and output, depending on the Resource Measurement Facility (RMF) options set, can be written to SMF data sets.

CICS Monitoring Facility (CMF) and the MVS workload manager

If you are running CICS with the MVS workload manager in *compatibility mode*, CICS supports the SYSEVENT class of monitoring by default, regardless of the status of the monitoring options. You do not need to set monitoring and sysevent monitoring on (with MN=ON and MNEVE=ON respectively).

If you are running CICS with the MVS workload manager environment in *goal mode*, the MVS workload manager provides transaction activity report reporting which replaces the SYSEVENT class of monitoring.

Using CICS monitoring SYSEVENT information with RMF

This section explains how to use the CICS monitoring facility with the Resource Measurement Facility (RMF) to obtain transaction rate reporting.

CICS usage of RMF transaction reporting

CICS monitoring facility with RMF provides a very useful tool for performing day-to-day monitoring of CICS transaction rates and response times.

The objective of using the CICS monitoring facility with RMF is to enable transaction rates and internal response times to be monitored without incurring the overhead of running the full CICS monitoring facility and associated reporting. This approach may be useful when only transaction statistics are required, rather than the very detailed information that CICS monitoring facility produces. An example of this is the monitoring of a production system where the minimum overhead is required.

CICS monitoring facility use of SYSEVENT

The CICS monitoring facility issues a SYSEVENT macro that gives to MVS/SRM the following information:

- The time at which the user-task was attached.
- The subsystem identification. This is derived from the first four characters of the CICS generic APPLID or from the four character name specified on the MNSUBSYS parameter if it is specified in the system initialization table (SIT).
- The transaction identifier of the task. This is the name of the CICS RDO transaction in the CICS program control table. This can be the name of a CICS system transaction, such as CSMI, CSNC, or CSPG.
- The user identifier.
- The specific APPLID of the CICS region. This is derived from the system initialization parameter, APPLID. It is expressed as the full 8 bytes of the transaction class parameter.

MVS IEAICS member

An IEAICS member needs to be coded and placed in SYS1.PARMLIB on the MVS system. For further information about this, see the *OS/390 MVS Initialization and Tuning Reference* manual. Reporting groups are assigned to the CICS system as a whole and to individual transactions.

How CMF SYSEVENT data is mapped to the IEAICSxx member of SYS1.PARMLIB

Table 4. How CMF SYSEVENT data is mapped to IEAICSxx

SYSEVENT macro	IEAICSxx member	CICS monitoring facility data
Transaction start time	N/A	Time at which user-task attached
Subsystem name	SUBSYS=	First 4 characters of the Generic APPLID
Transaction name	TRXNAME=	Transaction ID of task
User identification	USERID=	User ID
Transaction class	TRXCLASS=	The specific APPLID of the CICS region

For more information about how to use RMF, refer to the *MVS Resource Measurement Facility (RMF), Version 4.1.1 - Monitor I & II Reference and Users Guide*. If records are directed to SMF, refer to the *OS/390 MVS System Management Facilities (SMF)* manual. The following example shows the additional parameters that you need to add to your IEAICS member for two MRO CICS systems:

```
SUBSYS=ACIC,RPGN=100       /* CICS SYSTEM ACIC HAS REPORTING    */
  TRXNAME=CEMT,RPGN=101    /* GROUP OF 100 AND THERE ARE        */
  TRXNAME=USER,RPGN=102    /* THREE INDIVIDUAL GROUPS FOR       */
  TRXNAME=CSMI,RPGN=103    /* SEPARATE TRANSACTIONS             */
SUBSYS=BCIC,RPGN=200       /* CICS SYSTEM BCIC HAS REPORTING    */
  TRXNAME=CEMT,RPGN=201    /* GROUP OF 200 AND THERE ARE        */
  TRXNAME=USER,RPGN=202    /* THREE INDIVIDUAL GROUPS FOR       */
  TRXNAME=CSMI,RPGN=203    /* SEPARATE TRANSACTIONS             */
```

Notes:

1. The reporting group (number 100) assigned to the ACIC subsystem reports on all transactions in that system.
2. RMF reports on an individual transaction by name only if it is assigned a unique reporting group. If multiple transactions are defined with one reporting group, the name field is left blank in the RMF reports.

ERBRMF member for Monitor I session

This member defines the options that are used on the RMF Monitor I background session. This session does not include transaction reporting as used by CICS, but a Monitor I session has first to be active. A WKLD has to be defined to allow TRX reporting to be activated.

ERBRMF member for Monitor II session

This member defines the options that are used on the RMF Monitor II background session. This session performs transaction reporting as used by CICS. TRX defaults to TRX(ALLPGN) which reports on all transactions. Individual transactions can be named if desired.

RMF operations

A RMF job has to be started and this includes the Monitor I session. The RMF job should be started before initializing CICS. The RMF Monitor II session is started by

the command F RMF,S aa,MEMBER(xx) where 'aa' indicates alphabetic characters and 'xx' indicates alphanumeric characters.

Using the CICS monitoring facility with Tivoli Performance Reporter for OS/390

Tivoli Performance Reporter for OS/390 assists you in performance management and service-level management of a number of IBM products. The CICS Performance feature used by the Tivoli Performance Reporter provides reports for your use in analyzing the performance of CICS. See "Chapter 7. Tivoli Performance Reporter for OS/390" on page 109 for more information.

Event monitoring points

Product-sensitive programming interface

CICS monitoring data is collected at system-defined event monitoring points (EMPs) in the CICS code. Although you cannot relocate these monitoring points, you can choose which *classes* of monitoring data you want to be collected. Programming information about CICS monitoring is in the *CICS Customization Guide*.

If you want to gather more performance class data than is provided at the system-defined event monitoring points, you can code additional EMPs in your application programs. At these points, you can add or change up to 16384 bytes of user data in each performance record. Up to this maximum of 16384 bytes you can have, for each ENTRYNAME qualifier, any combination of the following:

- Between 0 and 256 counters
- Between 0 and 256 clocks
- A single 8192-byte character string.

You could use these additional EMPs to count the number of times a certain event occurs, or to time the interval between two events. If the performance class was active when a transaction was started, but was not active when a user EMP was issued, the operations defined in that user EMP would still execute on that transaction's monitoring area. The DELIVER option would result in a loss of data at this point, because the generated performance record cannot be output while the performance class is not active. If the performance class was not active when a transaction was started, the user EMP would have no effect.

User EMPs can use the EXEC CICS MONITOR command. For programming information about this command, refer to the *CICS Application Programming Reference*.

Additional EMPs are provided in some IBM program products, such as DL/I. From CICS's point of view, these are like any other user-defined EMP. EMPs in user applications and in IBM program products are identified by a decimal number. The numbers 1 through 199 are available for EMPs in user applications, and the numbers from 200 through 255 are for use in IBM program products. The numbers can be qualified with an 'entryname', so that you can use each number more than once. For example, PROGA.1, PROGB.1, and PROGC.1, identify three different EMPs because they have different entrynames.

For each user-defined EMP there must be a corresponding monitoring control table (MCT) entry, which has the same identification number and entryname as the EMP that it describes.

You do not have to assign entrynames and numbers to system-defined EMPs, and you do not have to code MCT entries for them.

Here are some ideas about how you might make use of the CICS and user fields provided with the CICS monitoring facility:

- If you want to time how long it takes to do a table lookup routine within an application, code an EMP with, say, ID=50 just before the table lookup routine and an EMP with ID=51 just after the routine. The system programmer codes a TYPE=EMP operand in the MCT for ID=50 to start user clock 1. You also code a TYPE=EMP operand for ID=51 to stop user clock 1. The application executes. When EMP 50 is processed, user clock 1 is started. When EMP 51 is processed, the clock is stopped.
- One user field could be used to accumulate an installation accounting unit. For example, you might count different amounts for different types of transaction. Or, in a browsing application, you might count 1 unit for each record scanned and not selected, and 3 for each record selected.

  You can also treat the fullword count fields as 32-bit flag fields to indicate special situations, for example, out-of-line situations in the applications, operator errors, and so on. CICS includes facilities to turn individual bits or groups of bits on or off in these counts.
- The performance clocks can be used for accumulating the time taken for I/O, DL/I scheduling, and so on. It usually includes any waiting for the transaction to regain control after the requested operation has completed. Because the periods are counted as well as added, you can get the average time waiting for I/O as well as the total. If you want to highlight an unusually long individual case, set a flag on in a user count as explained above.
- One use of the performance character string is for systems in which one transaction ID is used for widely differing functions. The application can enter a subsidiary ID into the string to indicate which particular variant of the transaction applies in each case.

  Some users have a single transaction ID so that all user input is routed through a common prologue program for security checking, for example. In this case, it is very easy to record the subtransaction identifier during this prologue. (However, it is equally possible to route transactions with different identifiers to the same program, in which case this technique is not necessary.)

End of Product-sensitive programming interface

The monitoring control table (MCT)

You use the monitoring control table (MCT):

- To tell CICS about the EMPs that you have coded in your application programs and about the data that is to be collected at these points
- To tell CICS that you want certain system-defined performance data not to be recorded during a particular CICS run.

DFHMCT TYPE=EMP

There must be a DFHMCT TYPE=EMP macro definition for every user-coded EMP. This macro has an ID operand, whose value must be made up of the

ENTRYNAME and POINT values specified on the EXEC CICS MONITOR command. The PERFORM operand of the DFHMCT TYPE=EMP macro tells CICS which user count fields, user clocks, and character values to expect at the identified user EMP, and what operations to perform on them.

DFHMCT TYPE=RECORD

The DFHMCT TYPE=RECORD macro allows you to *exclude* specific system-defined performance data from a CICS run. (Each performance monitoring record is approximately 1288 bytes long, without taking into account any user data that may be added, or any excluded fields.)

Each field of the performance data that is gathered at the system-defined EMPs belongs to a group of fields that has a group identifier. Each performance data field also has its own numeric identifier that is unique within the group identifier. For example, the transaction sequence number field in a performance record belongs to the group DFHTASK, and has the numeric identifier '031'. Using these identifiers, you can exclude specific fields or groups of fields, and reduce the size of the performance records.

Full details of the MCT are provided in the *CICS Resource Definition Guide*, and examples of MCT coding are included with the programming information in the *CICS Customization Guide*.

Three sample monitoring control tables are also provided in CICSTS13.CICS.SDFHSAMP:

- For terminal-owning regions (TORs) - DFHMCTT$
- For application-owning regions (AORs) - DFHMCTA$
- For application-owning regions (AORs) with DBCTL - DFHMCTD$
- For file-owning regions (FORs) - DFHMCTF$.

These samples show how to use the EXCLUDE and INCLUDE operands to reduce the size of the performance class record in order to reduce the volume of data written by CICS to SMF.

Controlling CICS monitoring

When CICS is initialized, you switch the monitoring facility on by specifying the system initialization parameter MN=ON. MN=OFF is the default setting. You can select the classes of monitoring data you want to be collected using the MNPER, MNEXC, and MNEVE system initialization parameters. You can request the collection of any combination of performance class data, exception class data, and SYSEVENT data. The class settings can be changed whether monitoring itself is ON or OFF. For guidance about system initialization parameters, refer to the *CICS System Definition Guide*.

When CICS is running, you can control the monitoring facility dynamically. Just as at CICS initialization, you can switch monitoring on or off, and you can change the classes of monitoring data that are being collected. There are two ways of doing this:

1. You can use the master terminal CEMT INQ|SET MONITOR command, which is described in the *CICS Supplied Transactions* manual.
2. You can use the EXEC CICS INQUIRE and SET MONITOR commands; programming information about these is in the *CICS System Programming Reference*.

If you activate a class of monitoring data in the middle of a run, the data for that class becomes available only for transactions that are started thereafter. You cannot change the classes of monitoring data collected for a transaction after it has started. It is often preferable, particularly for long-running transactions, to start all classes of monitoring data at CICS initialization.

Processing of CICS monitoring facility output

See "Chapter 7. Tivoli Performance Reporter for OS/390" on page 109 for more information.

Or, instead, you may want to write your own application program to process output from the CICS monitoring facility. The *CICS Customization Guide* gives programming information about the format of this output.

CICS provides a sample program, DFH$MOLS, which reads, formats, and prints monitoring data. It is intended as a sample program that you can use as a skeleton if you need to write your own program to analyze the data set. Comments within the program may help you if you want to do your own processing of CICS monitoring facility output. See the *CICS Operations and Utilities Guide* for further information on the DFH$MOLS program.

Performance implications

For information on the performance implications of using the CICS monitoring facility, see "CICS monitoring facility" on page 327.

Interpreting CICS monitoring

All of the exception class data and all of the system-defined performance class data that can be produced by CICS monitoring is listed below. Each of the data fields is presented as a field description, followed by an explanation of the contents. The field description has the format shown in Figure 4 on page 69, which is taken from the performance data group DFHTASK.

```
001 (TYPE-C, 'TRAN', 4 BYTES)
 |      |       |       |
 |      |       |       Length of the field (as re-
 |      |       |       presented by CMODLENG in the
 |      |       |       dictionary entry).
 |      |       |
 |      |      Informal name for the field, as used,
 |      |      perhaps, in column headings when the
 |      |      monitoring output is postprocessed
 |      |      (CMODHEAD of the dictionary entry).
 |      |
 |     Data type, which may be one of the following:
 |       A - a 32-bit count, a 64-bit count, a string of 64-bit counts
 |       C - a byte string
 |       P - a packed decimal value
 |       S - a clock comprising a 32-bit accumulation
 |           of 16-microsecond units followed by an
 |           8-bit flag followed by a 24-bit count
 |           (modulo-16 777 216) of the number of
 |           intervals included in the accumulation.
 |       T - a time stamp derived directly from the
 |           output of an STCK instruction.
 |      (CMODTYPE of the dictionary entry)
 |
Field identifier by which the field may be individually
excluded or included during MCT preparation (CMODIDNT of
the dictionary entry).
```

Figure 4. Format of the descriptions of the data fields

Note: References in Figure 4 to the associated dictionary entries apply only to the performance class data descriptions. Exception class data is not defined in the dictionary record.

Clocks and time stamps

In the descriptions that follow, the term *clock* is distinguished from the term *time stamp*.

A *clock* is a 32-bit value, expressed in units of 16 microseconds, accumulated during one or more measurement periods. The 32-bit value is followed by 8 reserved bits, which are in turn followed by a 24-bit value indicating the number of such periods.

Neither the 32-bit timer component of a clock nor its 24-bit period count are protected against wraparound. The timer capacity is about 18 hours, and the period count runs modulo 16 777 216.

The 8 reserved bits have the following significance:

Bits 0, 1, 2 and 3
Used for online control of the clock when it is running, and should always be zeros on output.

Bits 4 and 7
Not used.

Bits 5 and 6
Used to indicate, when set to 1, that the clock has suffered at least one out-of-phase start (bit 5) or stop (bit 6).

A *time stamp* is an 8-byte copy of the output of an STCK instruction.

Note: All times produced in the offline reports are in GMT (Greenwich Mean Time) not local time. Times produced by online reporting can be expressed in either GMT or local time.

Performance class data

The performance class data is described below in order of group name. The group name is always in field CMODNAME of the dictionary entry.

A user task can be represented by one or more performance class monitoring records, depending on whether the MCT event monitoring option DELIVER or the system initialization parameters MNCONV=YES or MNSYNC=YES have been selected. In the descriptions that follow, the term “user task” means “that part or whole of a transaction that is represented by a performance class record”, unless the description states otherwise.

A discussion about transaction timing fields

The CMF performance class record provides detailed timing information for each transaction as it is processed by CICS. A transaction can be represented by one or more performance class records depending on the monitoring options selected. The key transaction timing data fields are:

- The Transaction Start time and Stop time represent the start and end of a transaction measurement interval. This is normally the period between transaction attach and detach but the performance class record could represent a part of a transaction depending on the monitoring options selected. The "Transaction Response Time" can be calculated by subtracting the transaction start time from the stop time.
- The Transaction Dispatch time is the time the transaction was dispatched.
- The Transaction Dispatch Wait time is the time the transaction was suspended and waiting for redispatch.
- The Transaction CPU time is the portion of Dispatch time when the task is using processor cycles
- The Transaction Suspend time is the total time the task was suspended and includes:
  - All task suspend (wait) time, which includes:
    - The wait time for redispatch (dispatch wait)
    - The wait time for first dispatch (first dispatch delay)
    - The total I/O wait and other wait times.
- The First Dispatch Delay is then further broken down into:
  - First Dispatch Delay due to TRANCLASS limits
  - First Dispatch Delay due to MXT limits.

The CMF performance class record also provides a more detailed breakdown of the transaction suspend (wait) time into separate data fields. These include:

- Terminal I/O wait time
- File I/O wait time
- RLS File I/O wait time
- Journal I/O wait time
- Temporary Storage I/O wait time
- Shared Temporary Storage I/O wait time
- Inter-Region I/O wait time
- Transient Data I/O wait time
- LU 6.1 I/O wait time
- LU 6.2 I/O wait time

- FEPI suspend time
- Local ENQ delay time
- Global ENQ delay time
- RRMS/MVS Indoubt wait time
- Socket I/O wait time
- RMI suspend time
- Lock Manager delay time
- EXEC CICS WAIT EXTERNAL wait time
- EXEC CICS WAITCICS and WAIT EVENT wait time
- Interval Control delay time
- "Dispatchable Wait" wait time
- IMS(DBCTL) wait time
- DB2 ready queue wait time
- DB2 connection wait time
- DB2 wait time
- CFDT server syncpoint wait time
- Syncpoint delay time
- CICS BTS run process/activity synchronous wait time
- CICS MAXOPENTCBS delay time
- JVM suspend time

A note about response time

You can calculate the internal CICS response time by subtracting performance data field 005 (start time) from performance data field 006 (stop time).

Figure 5 shows the relationship of dispatch time, suspend time, and CPU time with the response time.

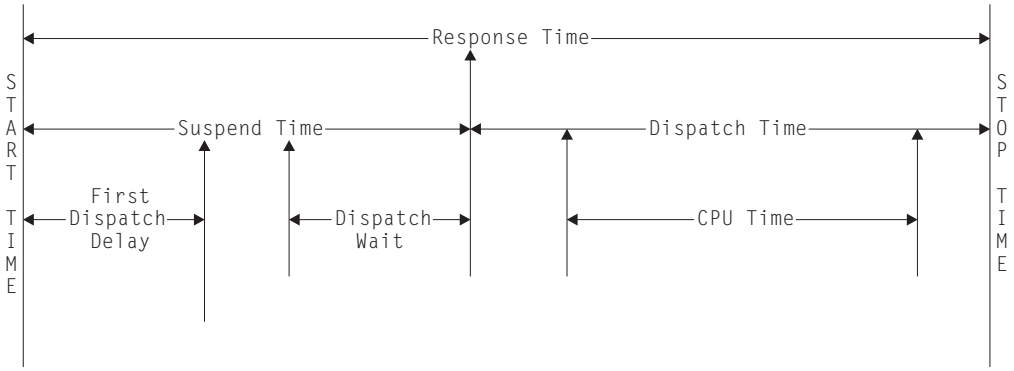


Figure 5. Response time relationships

A note about wait (suspend) times

The performance data fields 009, 010, 011, 063, 100, 101, 123, 128, 129, 133, 134, 156, 171, 174, 176, 177, 178, 181, 182, 183, 184, 186, 187, 188, 189, 191, 195, 196, 241, 250, and 254 all record the elapsed time spent waiting for a particular type of I/O operation. For example, field 009 records the elapsed time waiting for terminal I/O. The elapsed time includes not only that time during which the I/O operation is actually taking place, but also the time during which the access method is completing the outstanding event control block, and the time subsequent to that until the waiting CICS transaction is redispatched. See Table 5 on page 72 for the types of wait (suspend) fields. Figure 6 on page 73 shows an example of the relationship between a typical transaction wait time field, and the transaction's suspend time, dispatch time, CPU and dispatch wait time fields.

Table 5. Performance class wait (suspend) fields

Field-Id	Group Name	Description
009	DFHTERM	TC I/O wait time
010	DFHJOUR	JC I/O wait time
011	DFHTEMP	TS I/O wait time
063	DFHFILE	FC I/O wait time
100	DFHTERM	IR I/O wait time
101	DFHDEST	TD I/O wait time
123	DFHTASK	Global ENQ delay time
128	DFHTASK	Lock Manager delay time
129	DFHTASK	Local ENQ delay time
133	DFHTERM	TC I/O wait time - LU6.1
134	DFHTERM	TC I/O wait time - LU6.2
156	DFHFEPI	FEPI Suspend time
171	DFHTASK	Resource manager interface (RMI) suspend time
174	DFHFILE	RLS FC I/O wait time
176	DFHFILE	Coupling Facility data tables server I/O wait time
177	DFHSYNC	Coupling Facility data tables server syncpoint and resynchronization wait time
178	DFHTEMP	Shared TS I/O wait time
181	DFHTASK	EXEC CICS WAIT EXTERNAL wait time
182	DFHTASK	EXEC CICS WAITCICS and WAIT EVENT wait time
183	DFHTASK	Interval Control delay time
184	DFHTASK	"Dispatchable Wait" wait time
186	DFHDATA	IMS (DBCTL) wait time
187	DFHDATA	DB2 ready queue wait time
188	DFHDATA	DB2 connection time
189	DFHDATA	DB2 wait time
191	DFHTASK	RRMS/MVS wait time
195	DFHTASK	CICS BTS run process/activity synchronous wait time
196	DFHSYNC	Syncpoint delay time
241	DFHSOCK	Socket I/O wait time
249	DFHTASK	User task QR TCB wait-for-dispatch time
250	DFHTASK	CICS MAXOPENTCB delay time
254	DFHTASK	Java Virtual Machine (JVM) suspend time

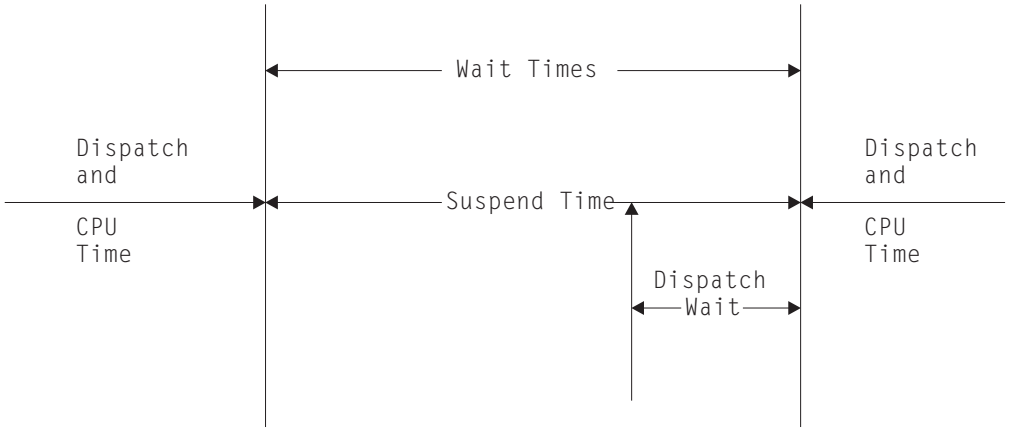


Figure 6. Wait (suspend) time relationships

Improvements to the CMF suspend time and wait time measurements allow you to perform various calculations on the suspend time accurately. For example, the "Total I/O Wait Time" can be calculated as follows:

Total I/O wait time =
(Terminal control I/O wait +
Temporary storage I/O wait +
Shared temporary storage I/O wait +
Transient data I/O wait +
Journal (MVS logger) I/O wait +
File control I/O wait +
RLS file I/O wait +
CF data table I/O wait +
Socket I/O wait +
Interregion (MRO) I/O wait +
LU 6.1 TC I/O wait +
LU 6.2 TC I/O wait +
FEPI I/O wait)

The "other wait time" (that is, uncaptured wait (suspend) time) can be calculated as follows:

Total other wait time =
(First dispatch delay +
Local ENQ delay +
Global ENQ delay +
Interval control delay +
Lock manager delay +
Wait external wait +
EXEC CICS WAITCICS and EXEC CICS WAIT EVENT wait +
CICS BTS run synchronous wait +
CFDT server synchronous wait +
Syncpoint delay time +

CICS MAXOPENTCBS delay +
RRMS/MVS wait +
RMI suspend +
JVM suspend time
"Dispatchable wait"s wait)

Note: The First Dispatch Delay performance class data field includes the MXT and TRANCLASS First Dispatch Delay fields.

The Uncaptured wait time can be calculated as follows:

Uncaptured wait time =

```
(Suspend - (total I/O wait time + total other wait time))
```

In addition to the transaction "Suspend (wait) Time" breakdown, the CMF performance class data provides several other important transaction timing measurements. They include:

- The Program load time is the program fetch time (dispatch time) for programs invoked by the transaction
- The Exception wait time is the accumulated time from the exception conditions as measured by the CMF exception class records. For more information, see "Exception class data" on page 102.
- The RMI elapsed time is the elapsed time the transaction spent in all Resource Managers invoked by the transaction using the Resource Manager Interface (RMI)
- The JVM elapsed time is the elapsed time the transaction spent in the Java Virtual Machine (JVM) for the the Java programs invoked by the transaction.
- The Syncpoint elapsed time is the elapsed time the transaction spent processing a syncpoint.

A note about program load time

Figure 7 shows the relationship between the program load time (field 115) and the

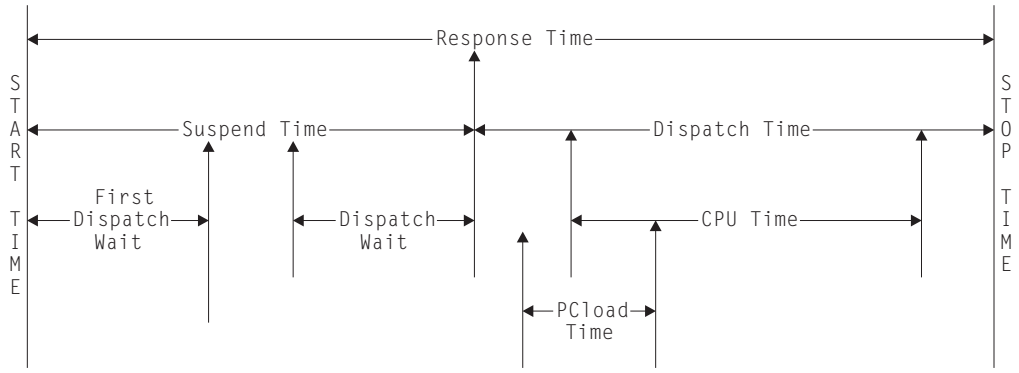


Figure 7. Program load time

dispatch time and the suspend time (fields 7 and 14).

A note about RMI elapsed and suspend time

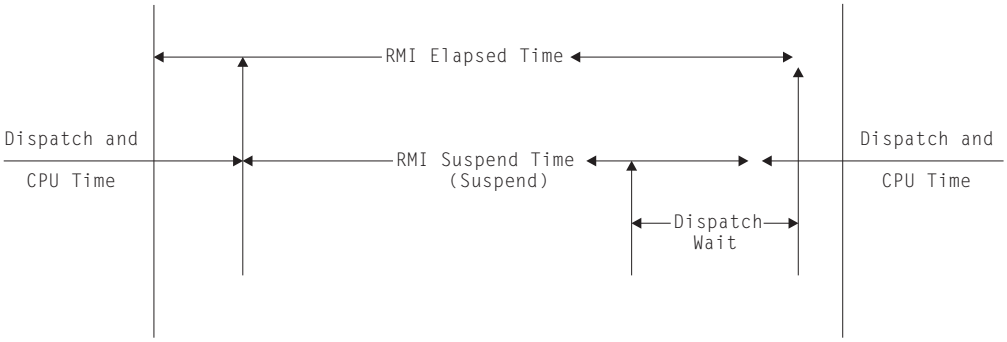


Figure 8. RMI elapsed and suspend time

Figure 8 shows the relationship between the RMI elapsed time and the suspend time (fields 170 and 171).

Note: In CICS Transaction Server for OS/390 Release 3, or later, the DB2 wait, the DB2 connection wait, and the DB2 readyq wait time fields as well as the IMS wait time field are included in the RMI suspend time.

JVM elapsed time and suspend time

The JVM elapsed and suspend time fields provide an insight into the amount of time that a transaction spends in a Java Virtual Machine (JVM).

Care must be taken when using the JVM elapsed time (group name DFHTASK, field id: 253) and JVM suspend time (group name DFHTASK, field id: 254) fields in any calculation with other CMF timing fields. This is because of the likelihood of double accounting other CMF timing fields in the performance class record within the JVM time fields. For example, if a Java application program invoked by a transaction issues a read file (non-RLS) request using the Java API for CICS (JCICS) classes, the file I/O wait time will be included in the both the file I/O wait time field (group name DFHFILE, field id: 063), the transaction suspend time field (group name DFHTASK, field id: 014) as well as the JVM suspend time field.

The JVM elapsed and suspend time fields are best evaluated from the overall transaction performance view and their relationship with the transaction response time, transaction dispatch time, and transaction suspend time. The performance class data also includes the amount of processor (CPU) time that a transaction used whilst in a JVM on a CICS J8 mode TCB in the J8CPUT field (group name: DFHTASK, field id: 260).

Note: The number of Java API for CICS (JCICS) requests issued by the user task is included in the CICS OO foundation class request count field (group name: DFHCICS, field id: 025).

A note about syncpoint elapsed time

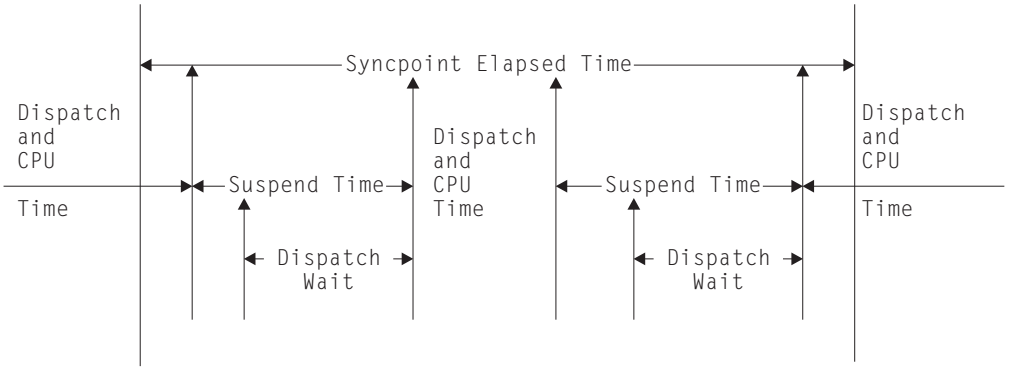


Figure 9. Syncpoint elapsed time

Figure 9 shows the relationship between the syncpoint elapsed time (field 173) and the suspend time (field 14).

A note about storage occupancy counts

An occupancy count measures the area under the curve of user-task storage in use against elapsed time. The unit of measure is the "byte-unit", where the "unit" is equal to 1024 microseconds, or 1.024 milliseconds. Where *ms* is milliseconds, a user task occupying, for example, 256 bytes for 125 milliseconds, is measured as follows:

125 / 1.024 *ms* = 122 units * 256 = 31 232 byte-units.

Note: All references to "Start time" and "Stop time" in the calculations below refer to the middle 4 bytes of each 8 byte start/stop time field. Bit 51 of Start time or Stop time represents a unit of 16 microseconds.

To calculate response time and convert into microsecond units:
Response = ((Stop time − Start time) * 16)
To calculate number of 1024 microsecond "units":
Units = (Response / 1024)
or
Units = ((Stop time − Start time) / 64)
To calculate the average user-task storage used from the storage occupancy count:
Average user-task storage used = (Storage Occupancy / Units)

To calculate units per second:
Units Per Second = (1 000 000 / 1024) = 976.5625
To calculate the response time in seconds:
Response time = (((Stop time − Start time) * 16) / 1 000 000)

During the life of a user task, CICS measures, calculates, and accumulates the storage occupancy at the following points:

- Before GETMAIN increases current user-storage values
- Before FREEMAIN reduces current user-storage values
- Just before the performance record is moved to the buffer.

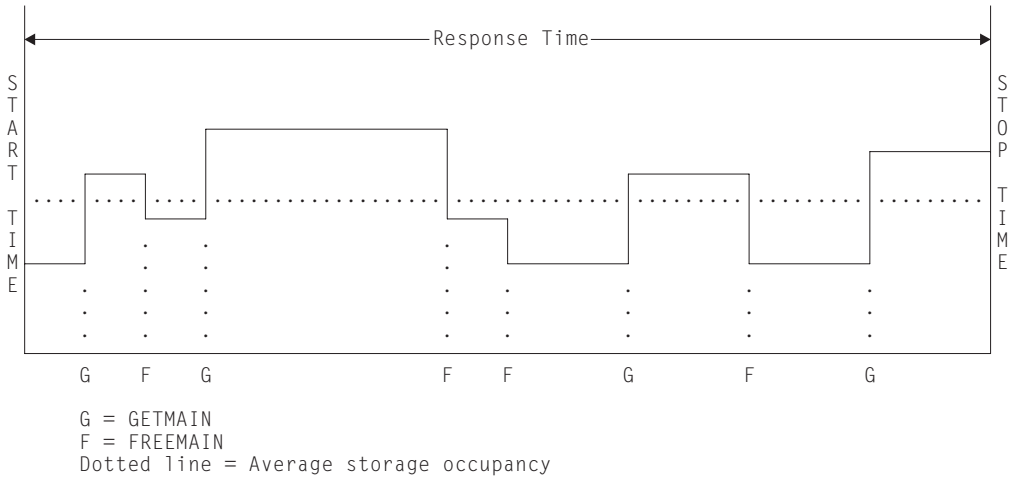


Figure 10. Storage occupancy

A note about program storage

The level of program storage currently in use is incremented at LOAD, LINK, and XCTL events by the size (in bytes) of the referenced program, and is decremented at RELEASE or RETURN events.

Note: On an XCTL event, the program storage currently in use is also decremented by the size of the program issuing the XCTL, because the program is no longer required.

Figure 11 on page 78 shows the relationships between the "high-water mark" data fields that contain the maximum amounts of program storage in use by the user task. Field PCSTGHWM (field ID 087) contains the maximum amount of program storage in use by the task both above *and* below the 16MB line. Fields PC31AHWM (139) and PC24BHWM (108) are subsets of PCSTGHWM, containing the maximum amounts in use above and below the 16MB line, respectively. Further subset-fields contain the maximum amounts of storage in use by the task in each of the CICS dynamic storage areas (DSAs).

Note: The totaled values of all the subsets in a superset may not necessarily equate to the value of the superset; for example, the value of PC31AHWM plus the value of PC24BHWM may not equal the value of PCSTGHWM. This is because the peaks in the different types of program storage acquired by the user task do not necessarily occur simultaneously.

The "high-water mark" fields are described in detail in "User storage fields in group DFHSTOR:" on page 87. For information about the program storage fields, see "Program storage fields in group DFHSTOR:" on page 89.

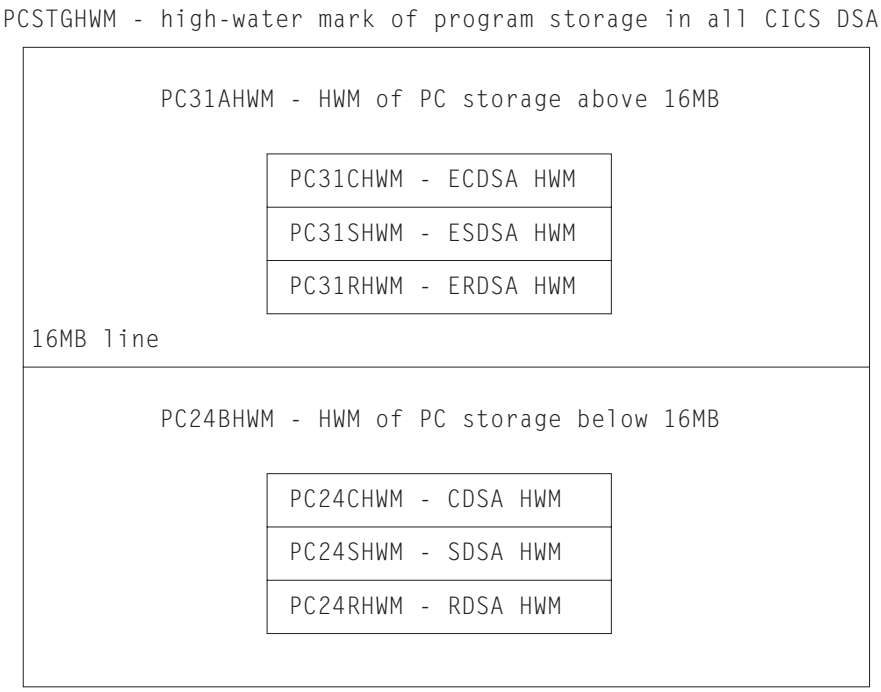


Figure 11. Relationships between the "high-water mark" program storage data fields

Performance data in group DFHCBTS

Group DFHCBTS contains the following performance data:

200 (TYPE-C, 'PRCSNAME', 36 BYTES)
The name of the CICS business transaction service (BTS) process of which the user task formed part.

201 (TYPE-C, 'PRCSTYPE', 8 BYTES)
The process-type of the CICS BTS process of which the user task formed part.

202 (TYPE-C, 'PRCSID', 52 BYTES)
The CICS-assigned identifier of the CICS BTS root activity that the user task implemented.

203 (TYPE-C, 'ACTVTYID', 52 BYTES)
The CICS-assigned identifier of the CICS BTS activity that the user task implemented.

204 (TYPE-C, 'ACTVTYNM', 16 BYTES)
The name of the CICS BTS activity that the user task implemented.

205 (TYPE-A, 'BARSYNCT', 4 BYTES)
The number of CICS BTS run process, or run activity, requests that the user task made in order to execute a process or activity synchronously.

206 (TYPE-A, 'BARASYCT', 4 BYTES)
The number of CICS BTS run process, or run activity, requests that the user task made in order to execute a process or activity asynchronously.

207 (Type-A, 'BALKPACT', 4 BYTES)
The number of CICS BTS link process, or link activity, requests that the user task issued.

208 (TYPE-A, 'BADPROCT', 4 BYTES)
The number of CICS BTS define process requests issued by the user task.

209 (TYPE-A, 'BADACTCT', 4 BYTES)
The number of CICS BTS define activity requests issued by the user task.

210 (TYPE-A, 'BARSPACT', 4 BYTES)
The number of CICS BTS reset process and reset activity requests issued by the user task.

211 (TYPE-A, 'BASUPACT', 4 BYTES)
The number of CICS BTS suspend process, or suspend activity, requests issued by the user task.

212 (TYPE-A, 'BARMPACT', 4 BYTES)
The number of CICS BTS resume process, or resume activity, requests issued by the user task.

213 (TYPE-A, 'BADCPACT', 4 BYTES)
The number of CICS BTS delete activity, cancel process, or cancel activity, requests issued by the user task.

214 (TYPE-A, 'BAACQPCT', 4 BYTES)
The number of CICS BTS acquire process, or acquire activity, requests issued by the user task.

215 (Type-A, 'BATOTPCT', 4 BYTES)
Total number of CICS BTS process and activity requests issued by the user task.

216 (TYPE-A, 'BAPRDCCT', 4 BYTES)
The number of CICS BTS delete, get, or put, container requests for process data containers issued by the user task.

217 (TYPE-A, 'BAACDCCT', 4 BYTES)
The number of CICS BTS delete, get, or put, container requests for current activity data containers issued by the user task.

218 (Type-A, 'BATOTCCT', 4 BYTES)
Total number of CICS BTS delete, get or put, process container and activity container requests issued by the user task.

219 (TYPE-A, 'BARATECT', 4 BYTES)
The number of CICS BTS retrieve-reattach event requests issued by the user task.

220 (TYPE-A, 'BADFIECT', 4 BYTES)
The number of CICS BTS define-input event requests issued by the user task.

221 (TYPE-A, 'BATIAECT', 4 BYTES)
The number of CICS BTS DEFINE TIMER EVENT, CHECK TIMER EVENT, DELETE TIMER EVENT, and FORCE TIMER EVENT requests issued by the user task.

222 (TYPE-A, 'BATOTECT', 4 BYTES)
Total number of CICS BTS event-related requests issued by the user task.

Performance data in group DFHCICS

Group DFHCICS contains the following performance data:

005 (TYPE-T, 'START', 8 BYTES)
Start time of measurement interval. This is one of the following:

- The time at which the user task was attached

- The time at which data recording was most recently reset in support of the MCT user event monitoring point DELIVER option or the monitoring options MNCONV, MNSYNC, or FREQUENCY.

For more information, see "Clocks and time stamps" on page 69.

Note: Response Time = STOP − START. For more information, see "A note about response time" on page 71.

006 (TYPE-T, 'STOP', 8 BYTES)
Finish time of measurement interval. This is either the time at which the user task was detached, or the time at which data recording was completed in support of the MCT user event monitoring point DELIVER option or the monitoring options MNCONV, MNSYNC or FREQUENCY. For more information, see "Clocks and time stamps" on page 69.

Note: Response Time = STOP − START. For more information, see "A note about response time" on page 71.

025 (TYPE-A, 'CFCAPICT', 4 BYTES)
Number of CICS OO foundation class requests, including the Java API for CICS (JCICS) classes, issued by the user task.

089 (TYPE-C, 'USERID', 8 BYTES)
User identification at task creation. This can also be the remote user identifier for a task created as the result of receiving an ATTACH request across an MRO or APPC link with attach-time security enabled.

103 (TYPE-S, 'EXWTTIME', 8 BYTES)
Accumulated data for exception conditions. The 32-bit clock contains the total elapsed time for which the user waited on exception conditions. The 24-bit period count equals the number of exception conditions that have occurred for this task. For more information, see "Exception class data" on page 102

Note: The performance class data field 'exception wait time' will be updated when exception conditions are encountered even when the exception class is inactive.

112 (TYPE-C, 'RTYPE', 4 BYTES)
Performance record type (low-order byte-3):
C Record output for a terminal converse
D Record output for a user EMP DELIVER request
F Record output for a long-running transaction
S Record output for a syncpoint
T Record output for a task termination.

130 (TYPE-C, 'RSYSID', 4 bytes)
The name (sysid) of the remote system to which this transaction was routed either statically or dynamically.

This field also includes the connection name (sysid) of the remote system to which this transaction was routed when using the CRTE routing transaction. The field will be null for those CRTE transactions which establish or cancel the transaction routing session.

Note: If the transaction was not routed or was routed locally, this field is set to null. Also see the program name (field 71).

131 (TYPE-A, 'PERRECNT', 4 bytes)
The number of performance class records written by the CICS Transaction Server for OS/390 Monitoring Facility (CMF) for the user task.

167 (TYPE-C, 'SRVCLASS', 8 bytes)
The MVS Workload Manager (WLM) service class for this transaction. This field is null if the transaction was WLM-classified in another CICS region.

168 (TYPE-C, 'RPTCLASS', 8 bytes)
The MVS Workload Manager (WLM) report class for this transaction. This field is null if the transaction was WLM-classified in another CICS region.

Performance data in group DFHDATA

Group DFHDATA contains the following performance data:

179 (TYPE-A, 'IMSREQCT', 4 bytes)
The number of IMS (DBCTL) requests issued by the user task.

180 (TYPE-A, 'DB2REQCT', 8 bytes)
The number of DB2 (EXEC SQL and IFI) requests issued by the user task.

186 (TYPE-S, 'IMSWAIT', 8 bytes)
The elapsed time in which the user task waited for DBCTL to service the IMS requests issued by the user task.

For more information, see "Clocks and time stamps" on page 69, and "A note about wait (suspend) times" on page 71.

Note: This field is a component of the task suspend time, SUSPTIME (014) field.

187 (TYPE-S, 'DB2RDYQW', 8 bytes)
The elapsed time in which the user task waited for a DB2 thread to become available.

For more information, see "Clocks and time stamps" on page 69, and "A note about wait (suspend) times" on page 71.

Note: This field is a component of the task suspend time, SUSPTIME (014) field.

188 (TYPE-S, 'DB2CONWT', 8 bytes)
The elapsed time in which the user task waited for a CICS DB2 subtask to become available.

For more information, see "Clocks and time stamps" on page 69, and "A note about wait (suspend) times" on page 71.

Note: This field is a component of the task suspend time, SUSPTIME (014) field.

189 (TYPE-S, 'DB2WAIT', 8 bytes)
The elapsed time in which the user task waited for DB2 to service the DB2 EXEC SQL and IFI requests issued by the user task.

For more information, see "Clocks and time stamps" on page 69, and "A note about wait (suspend) times" on page 71.

Note: This field is a component of the task suspend time, SUSPTIME (014) field.

Performance data in group DFHDEST

Group DFHDEST contains the following performance data:

041 (TYPE-A, 'TDGETCT', 4 BYTES)
Number of transient data GET requests issued by the user task.

042 (TYPE-A, 'TDPUTCT', 4 BYTES)
Number of transient data PUT requests issued by the user task.

043 (TYPE-A, 'TDPURCT', 4 BYTES)
Number of transient data PURGE requests issued by the user task.

091 (TYPE-A, 'TDTOTCT', 4 BYTES)
Total number of transient data requests issued by the user task. This field is the sum of TDGETCT, TDPUTCT, and TDPURCT.

101 (TYPE-S, 'TDIOWTT', 8 BYTES)
Elapsed time in which the user waited for VSAM transient data I/O. For more information see "Clocks and time stamps" on page 69, and "A note about wait (suspend) times" on page 71. This field is a subset of the task suspend time, SUSPTIME (014) field.

Performance data in group DFHDOCH

Group DFHDOCH contains the following performance data:

226 (TYPE-A, 'DHCRECT', 4 bytes)
The number of document handler CREATE requests issued by the user task.

227 (TYPE-A, 'DHINSCT', 4 bytes)
The number of document handler INSERT requests issued by the user task.

228 (TYPE-A, 'DHSETCT', 4 bytes)
The number of document handler SET requests issued by the user task.

229 (TYPE-A, 'DHRETCT', 4 bytes)
The number of document handler RETRIEVE requests issued by the user task.

230 (TYPE-A, 'DHTOTCT', 4 bytes)
The total number of document handler requests issued by the user task.

240 (TYPE-A, 'DHTOTDCL', 4 bytes)
The total length of all documents created by the user task.

Performance data in group DFHFEPI

Group DFHFEPI contains the following performance data:

150 (TYPE-A,'SZALLOCT', 4 bytes)
Number of conversations allocated by the user task. This number is incremented for each FEPI ALLOCATE POOL or FEPI CONVERSE POOL.

151 (TYPE-A,'SZRCVCT', 4 bytes)
Number of FEPI RECEIVE requests made by the user task. This number is also incremented for each FEPI CONVERSE request.

152 (TYPE-A,'SZSENDCT', 4 bytes)
Number of FEPI SEND requests made by the user task. This number is also incremented for each FEPI CONVERSE request.

153 (TYPE-A,'SZSTRTCT', 4 bytes)
Number of FEPI START requests made by the user task.

154 (TYPE-A,'SZCHROUT', 4 bytes)
Number of characters sent through FEPI by the user task.

155 (TYPE-A,'SZCHRIN', 4 bytes)
Number of characters received through FEPI by the user task.

156 (TYPE-S,'SZWAIT', 8 bytes)
Elapsed time in which the user task waited for all FEPI services. For more information see "Clocks and time stamps" on page 69, and "A note about wait (suspend) times" on page 71.

Note: This field is a component of the task suspend time, SUSPTIME (014) field.

157 (TYPE-A,'SZALLCTO', 4 bytes)
Number of times the user task timed out while waiting to allocate a conversation.

158 (TYPE-A,'SZRCVTO', 4 bytes)
Number of times the user task timed out while waiting to receive data.

159 (TYPE-A,'SZTOTCT', 4 bytes)
Total number of all FEPI API and SPI requests made by the user task.

Performance data in group DFHFILE

Group DFHFILE contains the following performance data:

036 (TYPE-A, 'FCGETCT', 4 BYTES)
Number of file GET requests issued by the user task.

037 (TYPE-A, 'FCPUTCT', 4 BYTES)
Number of file PUT requests issued by the user task.

038 (TYPE-A, 'FCBRWCT', 4 BYTES)
Number of file browse requests issued by the user task. This number excludes the START and END browse requests.

039 (TYPE-A, 'FCADDCT', 4 BYTES)
Number of file ADD requests issued by the user task.

040 (TYPE-A, 'FCDELCT', 4 BYTES)
Number of file DELETE requests issued by the user task.

063 (TYPE-S, 'FCIOWTT', 8 BYTES)
Elapsed time in which the user task waited for file I/O. For more information, see "Clocks and time stamps" on page 69, and "A note about wait (suspend) times" on page 71.

Note: This field is a component of the task suspend time, SUSPTIME (014) field.

070 (TYPE-A, 'FCAMCT', 4 BYTES)
Number of times the user task invoked file access-method interfaces. This number excludes requests for OPEN and CLOSE.

093 (TYPE-A, 'FCTOTCT', 4 BYTES)
Total number of file control requests issued by the user task. This number excludes any request for OPEN, CLOSE, ENABLE, or DISABLE of a file.

How EXEC CICS file commands correspond to file control monitoring fields is shown in Table 6.

Table 6. EXEC CICS file commands related to file control monitoring fields

EXEC CICS command	Monitoring fields
READ	FCGETCT and FCTOTCT
READ UPDATE	FCGETCT and FCTOTCT
DELETE (after READ UPDATE)	FCDELCT and FCTOTCT
DELETE (with RIDFLD)	FCDELCT and FCTOTCT
REWRITE	FCPUTCT and FCTOTCT
WRITE	FCADDCT and FCTOTCT
STARTBR	FCTOTCT
READNEXT	FCBRWCT and FCTOTCT
READNEXT UPDATE	FCBRWCT and FCTOTCT
READPREV	FCBRWCT and FCTOTCT
READPREV UPDATE	FCBRWCT and FCTOTCT
ENDBR	FCTOTCT
RESETBR	FCTOTCT
UNLOCK	FCTOTCT

Note: The number of STARTBR, ENDBR, RESETBR, and UNLOCK file control requests can be calculated by subtracting the file request counts, FCGETCT, FCPUTCT, FCBRWCT, FCADDCT, and FCDELCT from the total file request count, FCTOTCT.

174 (TYPE-S, 'RLSWAIT', 8 BYTES)
Elapsed time in which the user task waited for RLS file I/O. For more information, see "Clocks and time stamps" on page 69, and "A note about wait (suspend) times" on page 71.

Note: This field is a component of the task suspend time, SUSPTIME (014) field.

175 (TYPE-S, 'RLSCPUT', 8 BYTES)
The RLS File Request CPU (SRB) time field (RLSCPUT) is the SRB CPU time this transaction spent processing RLS file requests. This field should be added to the transaction CPU time field (USRCPUT) when considering the measurement of the total CPU time consumed by a transaction. Also, this field cannot be considered a subset of any other single CMF field (including RLSWAIT). This is because the RLS field requests execute asynchronously under an MVS SRB which can be running in parallel with the requesting transaction. It is also possible for the SRB to complete its processing before the requesting transaction waits for the RLS file request to complete.

Note: This clock field could contain a CPU time of zero with a count of greater than zero. This is because the CMF timing granularity is measured in 16 microsecond units and the RLS file request(s) may complete in less than that time unit.

176 (TYPE-S, 'CFDTWAIT', 8 BYTES)
Elapsed time in which the user task waited for a data table access request to

the Coupling Facility Data Table server to complete. For more information, see "Clocks and time stamps" on page 69, and "A note about wait (suspend) times" on page 71.

Note: This field is a component of the task suspend time, SUSPTIME (014) field.

Performance data in group DFHJOUR

Group DFHJOUR contains the following performance data:

010 (TYPE-S, 'JCIOWTT', 8 BYTES)
Elapsed time for which the user task waited for journal I/O. For more information, see "Clocks and time stamps" on page 69, and "A note about wait (suspend) times" on page 71.

Note: This field is a component of the task suspend time, SUSPTIME (014) field.

058 (TYPE-A, 'JNLWRTCT', 4 BYTES)
Number of journal write requests issued by the user task.

172 (TYPE-A, 'LOGWRTCT', 4 BYTES)
Number of CICS log stream write requests issued by the user task.

Performance data in group DFHMAPP

Group DFHMAPP contains the following performance data:

050 (TYPE-A, 'BMSMAPCT', 4 BYTES)
Number of BMS MAP requests issued by the user task. This field corresponds to the number of RECEIVE MAP requests that did not incur a terminal I/O, and the number of RECEIVE MAP FROM requests.

051 (TYPE-A, 'BMSINCT', 4 BYTES)
Number of BMS IN requests issued by the user task. This field corresponds to the number of RECEIVE MAP requests that incurred a terminal I/O.

052 (TYPE-A, 'BMSOUTCT', 4 BYTES)
Number of BMS OUT requests issued by the user task. This field corresponds to the number of SEND MAP requests.

090 (TYPE-A, 'BMSTOTCT', 4 BYTES)
Total number of BMS requests issued by the user task. This field is the sum of BMS RECEIVE MAP, RECEIVE MAP FROM, SEND MAP, SEND TEXT, and SEND CONTROL requests issued by the user task.

Performance data in group DFHPROG

Group DFHPROG contains the following performance data:

055 (TYPE-A, 'PCLINKCT', 4 BYTES)
Number of program LINK requests issued by the user task, including the link to the first program of the user task. This field does not include program LINK URM (user-replaceable module) requests.

056 (TYPE-A, 'PCXCTLCT', 4 BYTES)
Number of program XCTL requests issued by the user task.

057 (TYPE-A, 'PCLOADCT', 4 BYTES)
Number of program LOAD requests issued by the user task.

071 (TYPE-C, 'PGMNAME', 8 BYTES)
The name of the first program invoked at attach-time.

For a remote transaction:

- If this CICS definition of the remote transaction does not specify a program name, this field contains blanks.
- If this CICS definition of the remote transaction specifies a program name, this field contains the name of the specified program. (Note that this is not necessarily the program that is run on the remote system.)

For a dynamically-routed transaction, if the dynamic transaction routing program routes the transaction locally and specifies an alternate program name, this field contains the name of the alternate program.

For a dynamic program link (DPL) mirror transaction, this field contains the initial program name specified in the dynamic program LINK request. DPL mirror transactions can be identified using byte 1 of the transaction flags, TRANFLAG (164), field.

For an ONC RPC or WEB alias transaction, this field contains the initial application program name invoked by the alias transaction. ONC RPC or WEB alias transactions can be identified using byte 1 of the transaction flags, TRANFLAG (164), field.

072 (TYPE-A, 'PCLURMCT', 4 BYTES)
Number of program LINK URM (user-replaceable module) requests issued by, or on behalf of, the user task.

A user-replaceable module is a CICS-supplied program that is always invoked at a particular point in CICS processing, as if it were part of the CICS code. You can modify the supplied program by including your own logic, or replace it with a version that you write yourself.

The CICS-supplied user-replaceable modules are:

- bridge exit program
- program error program
- transaction restart program
- terminal error program
- node error program
- terminal autoinstall program(s)
- program autoinstall program
- dynamic routing program
- CICS-DBCTL interface status program
- CICS-DB2 dynamic plan exit program
- distributed dynamic routing program
- inbound IIOP exit program

For detailed information on CICS user-replaceable programs, see the *CICS Customization Guide*.

073 (TYPE-A, 'PCDPLCT', 4 BYTES)
Number of distributed program link (DPL) requests issued by the user task.

113 (TYPE-C, 'ABCODEO', 4 BYTES)
Original abend code.

114 (TYPE-C, 'ABCODEC', 4 BYTES)
Current abend code.

115 (TYPE-S, 'PCLOADTM', 8 BYTES)
Elapsed time in which the user task waited for program library (DFHRPL) fetches. Only fetches for programs with installed program definitions or autoinstalled as a result of application requests are included in this figure. However, installed programs residing in the LPA are not included (because they do not incur a physical fetch from a library). For more information about program load time, see "Clocks and time stamps" on page 69, and "A note about program load time" on page 74.

Performance data in group DFHSOCK

241 (TYPE-S, 'SOIOWTT', 8 BYTES)
The elapsed time in which the user task waited for socket I/O. For more information, see "Clocks and time stamps" on page 69, and "A note about wait (suspend) times" on page 71.

Note: This field is a component of the task suspend time, SUSPTIME (O14), field.

242 (TYPE-A, 'SOBYENCT', 4 BYTES)
The number of bytes encrypted by the secure sockets layer for the user task.

243 (TYPE-A, 'SOBYDECT', 4 BYTES)
The number of bytes decrypted by the secure sockets layer for the user task.

244 (TYPE-C, 'CLIPADDR', 16 BYTES)
The client IP address (*nnn.nnn.nnn.nnn*)

Performance data in group DFHSTOR

User storage fields in group DFHSTOR:

033 (TYPE-A, 'SCUSRHWM', 4 BYTES)
Maximum amount (high-water mark) of user storage allocated to the user task below the 16MB line, in the user dynamic storage area (UDSA).

054 (TYPE-A, 'SCUGETCT', 4 BYTES)
Number of user-storage GETMAIN requests issued by the user task below the 16MB line, in the UDSA.

095 (TYPE-A, 'SCUSRSTG', 8 BYTES)
Storage occupancy of the user task below the 16MB line, in the UDSA. This measures the area under the curve of storage in use against elapsed time. For more information about storage occupancy, see "A note about storage occupancy counts" on page 76.

105 (TYPE-A, 'SCUGETCT', 4 BYTES)
Number of user-storage GETMAIN requests issued by the user task for storage above the 16MB line, in the extended user dynamic storage area (EUDSA).

106 (TYPE-A, 'SCUSRHWM', 4 BYTES)
Maximum amount (high-water mark) of user-storage allocated to the user task above the 16MB line, in the EUDSA.

107 (TYPE-A, 'SCUCRSTG', 8 BYTES)
Storage occupancy of the user task above the 16MB line, in the EUDSA. This

measures the area under the curve of storage in use against elapsed time. For more information, see "A note about storage occupancy counts" on page 76.

116 (TYPE-A, 'SC24CHWM', 4 BYTES)
Maximum amount (high-water mark) of user-storage allocated to the user task below the 16MB line, in the CICS dynamic storage area (CDSA).

117 (TYPE-A, 'SCCGETCT', 4 BYTES)
Number of user-storage GETMAIN requests issued by the user task for storage below the 16MB line, in the CDSA.

118 (TYPE-A, 'SC24COCC', 8 BYTES)
Storage occupancy of the user task below the 16MB line, in the CDSA. This measures the area under the curve of storage in use against elapsed time. For more information, see "A note about storage occupancy counts" on page 76.

119 (TYPE-A, 'SC31CHWM', 4 BYTES)
Maximum amount (high-water mark) of user-storage allocated to the user task above the 16MB line, in the extended CICS dynamic storage area (ECDSA).

120 (TYPE-A, 'SCCGETCT', 4 BYTES)
Number of user-storage GETMAIN requests issued by the user task for storage above the 16MB line, in the ECDSA.

121 (TYPE-A, 'SC31COCC', 8 BYTES)
Storage occupancy of the user task above the 16MB line, in the ECDSA. This measures the area under the curve of storage in use against elapsed time. For more information, see "A note about storage occupancy counts" on page 76.

Table 7. User storage field id cross reference

	UDSA	EUDSA	CDSA	ECDSA
Getmain count	054	105	117	120
High-water-mark	033	106	116	119
Occupancy	095	107	118	121

Shared storage fields in group DFHSTOR:

144 (TYPE-A, 'SC24SGCT', 4 BYTES)
Number of storage GETMAIN requests issued by the user task for shared storage below the 16MB line, in the CDSA or SDSA.

145 (TYPE-A, 'SC24GSHR', 4 BYTES)
Number of bytes of shared storage GETMAINed by the user task below the 16MB line, in the CDSA or SDSA.

146 (TYPE-A, 'SC24FSHR', 4 BYTES)
Number of bytes of shared storage FREEMAINed by the user task below the 16MB line, in the CDSA or SDSA.

147 (TYPE-A, 'SC31SGCT', 4 BYTES)
Number of storage GETMAIN requests issued by the user task for shared storage above the 16MB line, in the ECDSA or ESDSA.

148 (TYPE-A, 'SC31GSHR', 4 BYTES)
Number of bytes of shared storage GETMAINed by the user task above the 16MB line, in the ECDSA or ESDSA.

149 (TYPE-A, 'SC31FSHR', 4 BYTES)
Number of bytes of shared storage FREEMAINed by the user task above the 16MB line, in the ECDSA or ESDSA.

Program storage fields in group DFHSTOR:

For more information on program storage see "Storage manager" on page 446.

087 (TYPE-A, 'PCSTGHWM', 4 BYTES)
Maximum amount (high-water mark) of program storage in use by the user task both above *and* below the 16MB line.

108 (TYPE-A, 'PC24BHWM', 4 BYTES)
Maximum amount (high-water mark) of program storage in use by the user task below the 16MB line. This field is a subset of PCSTGHWM (field ID 087) that resides below the 16MB line.

122 (TYPE-A, 'PC31RHWM', 4 BYTES)
Maximum amount (high-water mark) of program storage in use by the user task above the 16MB line, in the extended read-only dynamic storage area (ERDSA). This field is a subset of PC31AHWM (field ID 139) that resides in the ERDSA.

139 (TYPE-A, 'PC31AHWM', 4 BYTES)
Maximum amount (high-water mark) of program storage in use by the user task above the 16MB line. This field is a subset of PCSTGHWM (field ID 087) that resides above the 16MB line.

142 (TYPE-A, 'PC31CHWM', 4 BYTES)
Maximum amount (high-water mark) of program storage in use by the user task above the 16MB line, in the extended CICS dynamic storage area (ECDSA). This field is a subset of PC31AHWM (139) that resides in the ECDSA.

143 (TYPE-A, 'PC24CHWM', 4 BYTES)
Maximum amount (high-water mark) of program storage in use by the user task below the 16MB line, in the CICS dynamic storage area (CDSA). This field is a subset of PC24BHWM (108) that resides in the CDSA.

160 (TYPE-A, 'PC24SHWM', 4 BYTES)
Maximum amount (high-water mark) of program storage in use by the user task below the 16MB line, in the shared dynamic storage area (SDSA). This field is a subset of PC24BHWM (108) that resides in the SDSA.

161 (TYPE-A, 'PC31SHWM', 4 BYTES)
Maximum amount (high-water mark) of program storage in use by the user task above the 16MB line, in the extended shared dynamic storage area (ESDSA). This field is a subset of PC31AHWM (139) that resides in the ESDSA.

162 (TYPE-A, 'PC24RHWM', 4 BYTES)
Maximum amount (high-water mark) of program storage in use by the user task below the 16MB line, in the read-only dynamic storage area (RDSA). This field is a subset of PC24BHWM (108) that resides in the RDSA.

Performance data in group DFHSYNC

Group DFHSYNC contains the following performance data:

060 (TYPE-A, 'SPSYNCCT', 4 BYTES)
Number of SYNCPOINT requests issued during the user task.

Notes:

1. A SYNCPOINT is implicitly issued as part of the task-detach processing.
2. A SYNCPOINT is issued at PSB termination for DBCTL.

173 (TYPE-S, 'SYNCTIME', 8 BYTES)
Total elapsed time for which the user task was dispatched and was processing Syncpoint requests.

177 (TYPE-S, 'SRVSYWTT', 8 BYTES)
Total elapsed time in which the user task waited for syncpoint or resynchronization processing using the Coupling Facility data tables server to complete.

Note: This field is a component of the task suspend time, SUSPTIME (O14), field.

196 (TYPE-S, 'SYNCDLY', 8 BYTES)
The elapsed time in which the user task waited for a syncpoint request to be issued by it's parent transaction. The user task was executing as a result of the parent task issuing a CICS BTS run-process or run-activity request to execute a process or activity synchronously. For more information, see "Clocks and time stamps" on page 69, and "A note about wait (suspend) times" on page 71.

Note: This field is a component of the task suspend time, SUSPTIME (014) field.

Performance data in group DFHTASK

Group DFHTASK contains the following performance data:

001 (TYPE-C, 'TRAN', 4 BYTES)
Transaction identification.

004 (TYPE-C, 'TTYPE', 4 BYTES)
Transaction start type. The high-order bytes (0 and 1) are set to:
"TO" Attached from terminal input
"S" Attached by automatic transaction initiation (ATI) without data
"SD" Attached by automatic transaction initiation (ATI) with data
"QD" Attached by transient data trigger level
"U" Attached by user request
"TP" Attached from terminal TCTTE transaction ID
"SZ" Attached by Front End Programming Interface (FEPI).

007 (TYPE-S, 'USRDISPT', 8 BYTES)
Total elapsed time during which the user task was dispatched on each CICS TCB under which the task executed. This can include TCB modes QR, RO, CO, FO, SZ, RP, SL, SO, L8, J8, S8 and H8. For more information, see "Clocks and time stamps" on page 69.

008 (TYPE-S, 'USRCPUT', 8 BYTES)
Processor time for which the user task was dispatched on each CICS TCB under which the task executed. This can include TCB modes QR, RO, CO, FO, SZ, RP, SL, SO, L8, J8, S8 and H8. For more information see "Clocks and time stamps" on page 69.

014 (TYPE-S, 'SUSPTIME', 8 BYTES)
Total elapsed wait time for which the user task was suspended by the dispatcher. This includes:

- The elapsed time waiting for the first dispatch. This also includes any delay incurred because of the limits set for this transaction's transaction class (if any) or by the system parameter MXT being reached.
- The task suspend (wait) time.

- The elapsed time waiting for redispatch after a suspended task has been resumed.

For more information, see "A note about wait (suspend) times" on page 71.

031 (TYPE-P, 'TRANNUM', 4 BYTES)
Transaction identification number.

Note: The transaction number field is normally a 4-byte packed decimal number. However, some CICS system tasks are identified by special character 'transaction numbers', as follows:

- ' III' for system initialization task
- ' TCP' for terminal control.

These special identifiers are placed in bytes 2 through 4. Byte 1 is a blank (X'40') before the terminal control TCP identifier, and a null value (X'00') before the others.

059 (TYPE-A, 'ICPUINCT', 4 BYTES)
Number of interval control START or INITIATE requests during the user task.

064 (TYPE-A, 'TASKFLAG', 4 BYTES)
Task error flags, a string of 32 bits used for signaling unusual conditions occurring during the user task:

Bit 0 Reserved

Bit 1 Detected an attempt either to start a user clock that was already running, or to stop one that was not running

Bits 2–31
Reserved

066 (TYPE-A, 'ICTOTCT', 4 BYTES)
Total number of Interval Control Start, Cancel, Delay, and Retrieve requests issued by the user task.

082 (TYPE-C, 'TRNGRPID', 28 BYTES)
The transaction group ID is assigned at transaction attach time, and cab ve used to correlate the transactions that CICS executes for the same incoming work request (for example, the CWXN and CWBA transactions for Web requests). This transaction group ID relationship is useful when applied to the requests that originate through the CICS Web, IIOP, or 3270 bridge interface, as indicated by the transaction origin in Bytes 4 of the transaction flags field (group name DFHTASK, field ID 164).

097 (TYPE-C, 'NETUOWPX', 20 BYTES)
Fully qualified name by which the originating system is known to the VTAM network. This name is assigned at attach time using either the netname derived from the TCT (when the task is attached to a local terminal), or the netname passed as part of an ISC APPC or IRC attach header. At least three padding bytes (X'00') are present at the right end of the name.

If the originating terminal is VTAM across an ISC APPC or IRC link, the NETNAME is the *networkid.LUname*. If the terminal is non-VTAM, the NETNAME is *networkid.generic_applid*.

All originating information passed as part of an ISC LUTYPE6.1 attach header has the same format as the non-VTAM terminal originators above.

When the originator is communicating over an external CICS interface (EXCI) session, the name is a concatenation of:

```
'DFHEXCIU │ .       │ MVS Id    │ Address Space Id (ASID)'
 8 bytes  │ 1 byte  │ 4 bytes   │ 4 bytes
```

derived from the originating system. That is, the name is a 17-byte LU name consisting of:

- An 8-byte eye-catcher set to 'DFHEXCIU'.
- A 1-byte field containing a period (.).
- A 4-byte field containing the MVSID, in characters, under which the client program is running.
- A 4-byte field containing the address space id (ASID) in which the client program is running. This field contains the 4-character EBCDIC representation of the 2-byte hex address space id.

098 (TYPE-C, 'NETUOWSX', 8 BYTES)
Name by which the network unit of work id is known within the originating system. This name is assigned at attach time using either an STCK-derived token (when the task is attached to a local terminal), or the network unit of work id passed as part of an ISC APPC or IRC attach header.

The first six bytes of this field are a binary value derived from the system clock of the originating system and which can wrap round at intervals of several months.

The last two bytes of this field are for the period count. These may change during the life of the task as a result of syncpoint activity.

Note: When using MRO or ISC, the NETUOWSX field must be combined with the NETUOWPX field (097) to uniquely identify a task, because NETUOWSX is unique only to the originating CICS system.

102 (TYPE-S, 'DISPWTT', 8 BYTES)
Elapsed time for which the user task waited for redispatch. This is the aggregate of the wait times between each event completion and user-task redispatch.

Note: This field does not include the elapsed time spent waiting for first dispatch. This field is a component of the task suspend time, SUSPTIME (014), field.

109 (TYPE-C, 'TRANPRI', 4 BYTES)
Transaction priority when monitoring of the task was initialized (low-order byte-3).

Note: This field is a subset of the task suspend time, SUSPTIME (014), field.

123 (TYPE-S, 'GNQDELAY', 8 BYTES)
The elapsed time waiting for a CICS task control global enqueue. For more information, see "Clocks and time stamps" on page 69.

Note: This field is a subset of the task suspend time, SUSPTIME (014), field.

124 (TYPE-C, 'BRDGTRAN', 4 BYTES)
Bridge listener transaction identifier.

125 (TYPE-S, 'DSPDELAY', 8 BYTES)
The elapsed time waiting for first dispatch.

Note: This field is a component of the task suspend time, SUSPTIME (014), field. For more information, see “Clocks and time stamps” on page 69.

126 (TYPE-S, ‘TCLDELAY’, 8 BYTES)
The elapsed time waiting for first dispatch which was delayed because of the limits set for this transaction’s transaction class, TCLSNAME (166), being reached. For more information, see “Clocks and time stamps” on page 69.

Note: This field is a subset of the first dispatch delay, DSPDELAY (125), field.

127 (TYPE-S, ‘MXTDELAY’, 8 BYTES)
The elapsed time waiting for first dispatch which was delayed because of the limits set by the system parameter, MXT, being reached.

Note: The field is a subset of the first dispatch delay, DSPDELAY (125), field.

128 (TYPE-S, ‘LMDELAY’, 8 BYTES)
The elapsed time that the user task waited to acquire a lock on a resource. A user task cannot explicitly acquire a lock on a resource, but many CICS modules lock resources on behalf of user tasks using the CICS lock manager (LM) domain.

For more information about CICS lock manager, see *CICS Problem Determination Guide*.

For information about times, see “Clocks and time stamps” on page 69, and “A note about wait (suspend) times” on page 71.

Note: This field is a component of the task suspend time, SUSPTIME (014), field.

129 (TYPE-S, ‘ENQDELAY’, 8 BYTES)
The elapsed time waiting for a CICS task control local enqueue. For more information, see “Clocks and time stamps” on page 69.

Note: This field is a subset of the task suspend time, SUSPTIME (014), field.

132 (TYPE-C, ‘RMUOWID’, 8 BYTES)
The identifier of the unit of work (unit of recovery) for this task. Unit of recovery values are used to synchronize recovery operations among CICS and other resource managers, such as IMS and DB2.

163 (TYPE-C, ‘FCTYNAME’, 4 BYTES)
Transaction facility name. This field is null if the transaction is not associated with a facility. The transaction facility type (if any) can be identified using byte 0 of the transaction flags, TRANFLAG, (164) field.

164 (TYPE-A, ‘TRANFLAG’, 8 BYTES)
Transaction flags, a string of 64 bits used for signaling transaction definition and status information:

Byte 0 Transaction facility identification

Bit 0 Transaction facility name = none (x’80’)

Bit 1 Transaction facility name = terminal (x’40’)

If this Bit is set, FCTYNAME and TERM contain the same terminal id.

Bit 2 Transaction facility name = surrogate (x’20’)

Bit 3 Transaction facility name = destination (x'10')

Bit 4 Transaction facility name = 3270 bridge (x'08')

Bits 5–7
Reserved

Byte 1 Transaction identification information

Bit 0 System transaction (x'80')

Bit 1 Mirror transaction (x'40')

Bit 2 DPL mirror transaction (x'20')

Bit 3 ONC/RPC Alias transaction (x'10')

Bit 4 WEB Alias transaction (x'08')

Bit 5 3270 Bridge transaction (x'04')

Bit 6 Reserved (x'02')

Bit 7 CICS BTS Run transaction

Byte 2 MVS workload manager request (transaction) completion information

Bit 0 Report the total response time for completed work request (transaction)

Bit 1 Notify that the entire execution phase of the work request is complete

Bit 2 Notify that a subset of the execution phase of the work request is complete

Bits 3-7
Reserved

Byte 3 Transaction definition information

Bit 0 Taskdataloc = below (x'80')

Bit 1 Taskdatakey = cics (x'40')

Bit 2 Isolate = no (x'20')

Bit 3 Dynamic = yes (x'10')

Bits 4–7
Reserved

Byte 4 Transaction origin type

Byte 5 Reserved

Byte 6 Reserved

Byte 7 Recovery manager information

Bit 0 Indoubt wait = no

Bit 1 Indoubt action = commit

Bit 2 Recovery manager - UOW resolved with indoubt action

Bit 3 Recovery manager - Shunt

Bit 4 Recovery manager - Unshunt

Bit 5 Recovery manager - Indoubt failure

Bit 6 Recovery manager - Resource owner failure

Bit 7 Reserved

Note: Bits 2 through 6 will be reset on a SYNCPOINT request when the MNSYNC=YES option is specified.

166 (TYPE-C, 'TCLSNAME', 8 BYTES)
Transaction class name. This field is null if the transaction is not in a TRANCLASS.

170 (TYPE-S, 'RMITIME', 8 BYTES)
Amount of elapsed time spent in the Resource Manager Interface (RMI). For more information, see "Clocks and time stamps" on page 69, "A note about wait (suspend) times" on page 71, and Figure 8 on page 75.

171 (TYPE-S, 'RMISUSP', 8 BYTES)
Amount of elapsed time the task was suspended by the dispatcher while in the Resource Manager Interface (RMI). For more information, see "Clocks and time stamps" on page 69, "A note about wait (suspend) times" on page 71, and Figure 8 on page 75.

Note: The field is a subset of the task suspend time, SUSPTIME (014), field and also the RMITIME (170) field.

181 (TYPE-S, 'WTEXWAIT', 8 BYTES)
The elapsed time that the user task waited for one or more ECBs, passed to CICS by the user task using the EXEC CICS WAIT EXTERNAL ECBLIST command, to be MVS POSTed. The user task can wait on one or more ECBs. If it waits on more than one, it is dispatchable as soon as one of the ECBs is posted. For more information, see "Clocks and time stamps" on page 69, and "A note about wait (suspend) times" on page 71.

Note: This field is a component of the task suspend time, (SUSPTIME) (014), field.

182 (TYPE-S, 'WTCEWAIT', 8 BYTES)
The elapsed time the user task waited for:

- One or more ECBs, passed to CICS by the user task using the EXEC CICS WAITCICS ECBLIST command, to be MVS POSTed. The user task can wait on one or more ECBs. If it waits on more than one, it is dispatchable as soon as one of the ECBs is posted.
- Completion of an event initiated by the same or by another user task. The event would normally be the posting, at the expiration time, of a timer-event control area provided in response to an EXEC CICS POST command. The EXEC CICS WAIT EVENT command provides a method of directly giving up control to some other task until the event being waited on is completed.

For more information, see "Clocks and time stamps" on page 69, and "A note about wait (suspend) times" on page 71.

Note: This field is a component of the task suspend time, SUSPTIME (014), field.

183 (TYPE-S, 'ICDELAY', 8 BYTES)
The elapsed time the user task waited as a result of issuing either:

- An interval control EXEC CICS DELAY command for a specified time interval, or
- A specified time of day to expire, or

- An interval control EXEC CICS RETRIEVE command with the WAIT option specified. For more information, see "Clocks and time stamps" on page 69, and "A note about wait (suspend) times" on page 71.

Note: This field is a component of the task suspend time, SUSPTIME (014), field.

184 (TYPE-S, 'GVUPWAIT', 8 BYTES)
The elapsed time the user task waited as a result of giving up control to another task. A user task can give up control in many ways. Some examples are application programs that use one or more of the following EXEC CICS API or SPI commands:

- Using the EXEC CICS SUSPEND command. This command causes the issuing task to relinquish control to another task of higher or equal dispatching priority. Control is returned to this task as soon as no other task of a higher or equal priority is ready to be dispatched.
- Using the EXEC CICS CHANGE TASK PRIORITY command. This command immediately changes the priority of the issuing task and causes the task to give up control in order for it to be dispatched at its new priority. The task is not redispatched until tasks of higher or equal priority, and that are also dispatchable, have been dispatched.
- Using the EXEC CICS DELAY command with INTERVAL (0). This command causes the issuing task to relinquish control to another task of higher or equal dispatching priority. Control is returned to this task as soon as no other task of a higher or equal priority is ready to be dispatched.
- Using the EXEC CICS POST command requesting notification that a specified time has expired. This command causes the issuing task to relinquish control to give CICS the opportunity to post the time-event control area.
- Using the EXEC CICS PERFORM RESETTIME command to synchronize the CICS date and time with the MVS system date and time of day.
- Using the EXEC CICS START TRANSID command with the ATTACH option.

For more information, see "Clocks and time stamps" on page 69, and "A note about wait (suspend) times" on page 71.

Note: This field is a component of the task suspend time, SUSPTIME (014), field.

190 (TYPE-C, 'RRMSURID', 16 BYTES)
RRMS/MVS unit-of-recovery ID (URID).

191 (TYPE-S, 'RRMSWAIT', 8 BYTES)
The elapsed time in which the user task waited indoubt using resource recovery services for EXCI.

For more information, see "Clocks and time stamps" on page 69, and "A note about wait (suspend) times" on page 71.

Note: This field is a component of the task suspend time, SUSPTIME (014), field.

195 (TYPE-S, 'RUNTRWTT', 8 BYTES)
The elapsed time in which the user task waited for completion of a

transaction that executed as a result of the user task issuing a CICS BTS run process, or run activity, request to execute a process, or activity, synchronously.

For more information, see "Clocks and time stamps" on page 69, and "A note about wait (suspend) times" on page 71.

Note: This field is a component of the task suspend time, SUSPTIME (014), field.

248 (TYPE-A, 'CHMODECT', 4 BYTES)
The number of CICS change-TCB modes issued by the user task.

249 (TYPE-S, 'QRMODDLY', 8 BYTES)
The elapsed time for which the user task waited for redispatch on the CICS QR TCB. This is the aggregate of the wait times between each event completion. and user-task redispatch.

Note: This field does not include the elapsed time spent waiting for the first dispatch. The QRMODDLY field is a component of the task suspend time, SUSPTIME (014), field.

250 (TYPE-S, 'MXTOTDLY', 8 BYTES)
The elapsed time in which the user task waited to obtain a CICS open TCB, because the region had reached the limit set by the system parameter, MAXOPENTCBS.

For more information, see "Clocks and time stamps" on page 69, and "A note about wait (suspend) times" on page 71.

Note: This field is a component of the task suspend time,

251 (TYPE-A, 'TCBATTCT', 4 BYTES)
The number of CICS TCBs attached by or on behalf of the user task.

253 (TYPE-S, 'JVMTIME', 8 BYTES)
The elapsed time spent in the CICS JVM by the user task

254 (TYPE-S, 'JVMSUSP', 8 BYTES)
The elapsed time the user task was suspended by the CICS dispatcher while running in the CICS JVM.

Note: This field is a subset of the task suspend time, SUSPTIME (014), field.

255 (TYPE-S, 'QRDISPT', 8 BYTES)
The elapsed time for which the user task was dispatched on the CICS QR TCB. For more information, see "Clocks and time stamps" on page 69.

256 (TYPE-S, 'QRCPUT', 8 BYTES)
The processor time for which the user task was dispatched on the CICS QR TCB. For more information, see "Clocks and time stamps" on page 69.

257 (TYPE-S, 'MSDISPT', 8 BYTES)
Elapsed time for which the user task was dispatched on each CICS TCB (RO, CO, FO, SZ if FEPI is active, and RP if the ONC/RPC or CICS WEB Interface Feature is installed and active. Modes SO and SL are used only if TCPIP=YES is specified as a system initialization parameter). For more information, see "Clocks and time stamps" on page 69.

258 (TYPE-S, 'MSCPUT', 8 BYTES)
The processor time for which the user task was dispatched on each CICS TCB (RO, CO, FO, SZ if FEPI is active, and RP if the ONC/RPC or CICS WEB interface feature is installed and active. Modes SO and SL are used only if TCPIP=YES is specified as a system initialization parameter). For more information, see "Clocks and time stamps" on page 69.

259 (TYPE-S, 'L8CPUT', 8 BYTES)
The processor time for which the user task was dispatched on the CICS L8 TCB. For more information see "Clocks and time stamps" on page 69.

260 (TYPE-S, 'J8CPUT', 8 BYTES)
The processor time for which the user task was dispatched on each CICS J8 TCB. For more information, see "Clocks and time stamps" on page 69.

261 (TYPE-S, 'S8CPUT', 8 BYTES)
The processor time for which the user task was dispatched on the CICS S8 TCB. For more information, see "Clocks and time stamps" on page 69.

Performance data in group DFHTEMP

Group DFHTEMP contains the following performance data:

011 (TYPE-S, 'TSIOWTT', 8 BYTES)
Elapsed time for which the user task waited for VSAM temporary storage I/O. For more information see "Clocks and time stamps" on page 69, and "A note about wait (suspend) times" on page 71.

Note: This field is a component of the task suspend time, SUSPTIME (014), field.

044 (TYPE-A, 'TSGETCT', 4 BYTES)
Number of temporary-storage GET requests issued by the user task.

046 (TYPE-A, 'TSPUTACT', 4 BYTES)
Number of PUT requests to auxiliary temporary storage issued by the user task.

047 (TYPE-A, 'TSPUTMCT', 4 BYTES)
Number of PUT requests to main temporary storage issued by the user task.

092 (TYPE-A, 'TSTOTCT', 4 BYTES)
Total number of temporary storage requests issued by the user task. This field is the sum of the temporary storage READQ (TSGETCT), WRITEQ AUX (TSPUTACT), WRITEQ MAIN (TSPUTMCT), and DELETEQ requests issued by the user task.

178 (TYPE-S, 'TSSHWAIT', 8 BYTES)
Elapsed time that the user task waited for an asynchronous shared temporary storage request to a temporary storage data server to complete. For more information, see "Clocks and time stamps" on page 69, and "A note about wait (suspend) times" on page 71.

Note: This field is a component of the task suspend time, SUSPTIME (014), field.

Performance data in group DFHTERM

Group DFHTERM contains the following performance data:

002 (TYPE-C, 'TERM', 4 BYTES)
Terminal or session identification. This field is null if the task is not associated with a terminal or session.

009 (TYPE-S, 'TCIOWTT', 8 BYTES)
Elapsed time for which the user task waited for input from the terminal operator, after issuing a RECEIVE request. For more information, see "Clocks and time stamps" on page 69, and "A note about wait (suspend) times" on page 71.

Note: This field is a component of the task suspend time, SUSPTIME (014), field.

034 (TYPE-A, 'TCMSGIN1', 4 BYTES)
Number of messages received from the task's principal terminal facility, including LUTYPE6.1 and LUTYPE6.2 (APPC) but not MRO (IRC).

035 (TYPE-A, 'TCMSGOU1', 4 BYTES)
Number of messages sent to the task's principal terminal facility, including LUTYPE6.1 and LUTYPE6.2 (APPC) but not MRO (IRC).

067 (TYPE-A, 'TCMSGIN2', 4 BYTES)
Number of messages received from the LUTYPE6.1 alternate terminal facilities by the user task.

068 (TYPE-A, 'TCMSGOU2', 4 BYTES)
Number of messages sent to the LUTYPE6.1 alternate terminal facilities by the user task.

069 (TYPE-A, 'TCALLOCT', 4 BYTES)
Number of TCTTE ALLOCATE requests issued by the user task for LUTYPE6.2 (APPC), LUTYPE6.1, and IRC sessions.

083 (TYPE-A, 'TCCHRIN1', 4 BYTES)
Number of characters received from the task's principal terminal facility, including LUTYPE6.1 and LUTYPE6.2 (APPC) but not MRO (IRC).

084 (TYPE-A, 'TCCHROU1', 4 BYTES)
Number of characters sent to the task's principal terminal facility, including LUTYPE6.1 and LUTYPE6.2 (APPC) but not MRO (IRC).

085 (TYPE-A, 'TCCHRIN2', 4 BYTES)
Number of characters received from the LUTYPE6.1 alternate terminal facilities by the user task. *(Not applicable to ISC APPC.)*

086 (TYPE-A, 'TCCHROU2', 4 BYTES)
Number of characters sent to the LUTYPE6.1 alternate terminal facilities by the user task. *(Not applicable to ISC APPC.)*

100 (TYPE-S, 'IRIOWTT', 8 BYTES)
Elapsed time for which the user task waited for control at this end of an MRO link. For more information, see "Clocks and time stamps" on page 69, and "A note about wait (suspend) times" on page 71.

Note: This field is a component of the task suspend time, SUSPTIME (014), field.

111 (TYPE-C, 'LUNAME', 8 BYTES)
VTAM logical unit name (if available) of the terminal associated with this transaction. If the task is executing in an application-owning or file-owning region, the LUNAME is the generic applid of the originating connection for

MRO, LUTYPE6.1, and LUTYPE6.2 (APPC). The LUNAME is blank if the originating connection is an external CICS interface (EXCI).

133 (TYPE-S, 'LU61WTT', 8 BYTES)
The elapsed time for which the user task waited for I/O on a LUTYPE6.1 connection or session. This time also includes the waits incurred for conversations across LUTYPE6.1 connections, but not the waits incurred due to LUTYPE6.1 syncpoint flows. For more information see "Clocks and time stamps" on page 69, and "A note about wait (suspend) times" on page 71.

Note: This field is a component of the task suspend time, SUSPTIME (014), field.

134 (TYPE-S, 'LU62WTT', 8 BYTES)
The elapsed time for which the user task waited for I/O on a LUTYPE6.2 (APPC) connection or session. This time also includes the waits incurred for conversations across LUTYPE6.2 (APPC) connections, but not the waits incurred due to LUTYPE6.2 (APPC) syncpoint flows. For more information, see "Clocks and time stamps" on page 69, and "A note about wait (suspend) times" on page 71

Note: This field is a component of the task suspend time, SUSPTIME (014), field.

135 (TYPE-A, 'TCM62IN2', 4 BYTES)
Number of messages received from the alternate facility by the user task for LUTYPE6.2 (APPC) sessions.

136 (TYPE-A, 'TCM62OU2', 4 BYTES)
Number of messages sent to the alternate facility by the user task for LUTYPE6.2 (APPC) sessions.

137 (TYPE-A, 'TCC62IN2', 4 BYTES)
Number of characters received from the alternate facility by the user task for LUTYPE6.2 (APPC) sessions.

138 (TYPE-A, 'TCC62OU2', 4 BYTES)
Number of characters sent to the alternate facility by the user task for LUTYPE6.2 (APPC) sessions.

165 (TYPE-A, 'TERMINFO', 4 BYTES)
Terminal or session information for this task's principal facility as identified in the 'TERM' field id 002. This field is null if the task is not associated with a terminal or session facility.

Byte 0 Identifies whether this task is associated with a terminal or session. This field can be set to one of the following values:
X'00' None
X'01' Terminal
X'02' Session

Byte 1 If the principal facility for this task is a session (Byte 0 = x'02'), this field identifies the session type. This field can be set to one of the following values:
X'00' None
X'01' IRC
X'02' IRC XM
X'03' IRC XCF
X'04' LU61
X'05' LU62 Single

X'06' LU62 Parallel

Byte 2 Identifies the access method defined for the terminal id or session id in field TERM. This field can be set to one of the following values:
X'00' None
X'01' VTAM
X'02' BTAM
X'03' BSAM
X'04' TCAM
X'05' TCAMSNA
X'06' BGAM
X'07' CONSOLE

Byte 3 Identifies the terminal or session type for the terminal id or session id in TERM.

- See RDO Typeterm

For a list of the typeterm definitions, see the *CICS Resource Definition Guide*.

169 (TYPE-C, 'TERMCNNM', 4 BYTES)
Terminal session connection name. If the terminal facility associated with this transaction is a session, this field is the name of the owning connection (sysid).

A terminal facility can be identified as a session by using byte 0 of the terminal information, TERMINFO (165), field. If the value is x'02' the terminal facility is a session.

Performance data in group DFHWEBB

Group DFHWEBB contains the following performance data:

231 (TYPE-A, 'WBRCVCT', 4 BYTES)
The number of CICS Web interface RECEIVE requests issued by the user task.

232 (TYPE-A, 'WBCHRIN', 4 BYTES)
The number of characters received by the CICS Web interface RECEIVE requests issued by the user task.

233 (TYPE-A, 'WBSENDCT', 4 BYTES)
The number of CICS Web interface SEND requests issued by the user task.

234 (TYPE-A, 'WBCHROUT', 4 BYTES)
The number of characters sent by the CICS Web interface SEND requests issued by the user task.

235 (TYPE-A, 'WBTOTWCT', 4 BYTES)
The total number of CICS Web interface requests issued by the user task.

236 (TYPE-A, 'WBREPRCT', 4 BYTES)
The number of reads from the repository in shared temporary storage issued by the user task.

237 (TYPE-A, 'WBREPWCT', 4 BYTES)
The number of writes to the repository in shared temporary storage issued by the user task.

Exception class data

Exception records are produced after each of the following conditions encountered by a transaction has been resolved:

- Wait for storage in the CDSA
- Wait for storage in the UDSA
- Wait for storage in the SDSA
- Wait for storage in the RDSA
- Wait for storage in the ECDSA
- Wait for storage in the EUDSA
- Wait for storage in the ESDSA
- Wait for storage in the ERDSA
- Wait for auxiliary temporary storage
- Wait for auxiliary temporary storage string
- Wait for auxiliary temporary storage buffer
- Wait for coupling facility data tables locking (request) slot
- Wait for coupling facility data tables non-locking (request) slot (With coupling facility data tables each CICS has a number of slots available for requests in the CF data table. When all available slots are in use, any further request must wait.)
- Wait for file buffer
- Wait for file string
- Wait for LSRPOOL buffer
- Wait for LSRPOOL string

These records are fixed format. The format of these exception records is as follows:

```
MNEXCDS  DSECT
EXCMNTRN DS    CL4               TRANSACTION IDENTIFICATION
EXCMNTER DS    XL4               TERMINAL IDENTIFICATION
EXCMNUSR DS    CL8               USER IDENTIFICATION
EXCMNTST DS    CL4               TRANSACTION START TYPE
EXCMNSTA DS    XL8               EXCEPTION START TIME
EXCMNSTO DS    XL8               EXCEPTION STOP TIME
EXCMNTNO DS    PL4               TRANSACTION NUMBER
EXCMNTPR DS    XL4               TRANSACTION PRIORITY
         DS    CL4               RESERVED
EXCMNLUN DS    CL8               LUNAME
         DS    CL4               RESERVED
EXCMNEXN DS    XL4               EXCEPTION NUMBER
EXCMNRTY DS    CL8               EXCEPTION RESOURCE TYPE
EXCMNRID DS    CL8               EXCEPTION RESOURCE ID
EXCMNTYP DS    XL2               EXCEPTION TYPE
EXCMNWT  EQU   X'0001'             WAIT
EXCMNBWT EQU   X'0002'             BUFFER WAIT
EXCMNSWT EQU   X'0003'             STRING WAIT
         DS    CL2               RESERVED
EXCMNTCN DS    CL8               TRANSACTION CLASS NAME
EXCMNSRV DS    CL8               SERVICE CLASS NAME
EXCMNRPT DS    CL8               REPORT CLASS NAME
EXCMNNPX DS    CL20              NETWORK UNIT_OF_WORK PREFIX
EXCMNNSX DS    XL8               NETWORK UNIT_OF_WORK SUFFIX
EXCMNTRF DS    XL8               TRANSACTION FLAGS
EXCMNFCN DS    CL4               TRANSACTION FACILITY NAME
EXCMNCPN DS    CL8               CURRENT PROGRAM NAME
EXCMNBTR DS    CL4               BRIDGE TRANSACTION ID
```

```
| EXCMNURI DS    XL16              MVS/RRMS Unit of Recovery Id
| EXCMNRIL DS    F                 EXCEPTION RESOURCE ID LENGTH
| EXCMNRIX DS    XL256             EXCEPTION RESOURCE ID (EXTENDED)
*        END OF EXCEPTION RECORD ...
```

Exception data field descriptions

EXCMNTRN (TYPE-C, 4 BYTES)
Transaction identification.

EXCMNTER (TYPE-C, 4 BYTES)
Terminal identification. This field is null if the task is not associated with a terminal or session.

EXCMNUSR (TYPE-C, 8 BYTES)
User identification at task creation. This can also be the remote user identifier for a task created as the result of receiving an ATTACH request across an MRO or APPC link with attach-time security enabled.

EXCMNTST (TYPE-C, 4 BYTES)
Transaction start type. The low-order byte (0 and 1) is set to:

"TO" Attached from terminal input
"S" Attached by automatic transaction initiation (ATI) without data
"SD" Attached by automatic transaction initiation (ATI) with data
"QD" Attached by transient data trigger level
"U" Attached by user request
"TP" Attached from terminal TCTTE transaction ID
"SZ" Attached by Front End Programming Interface (FEPI)

EXCMNSTA (TYPE-T, 8 BYTES)
Start time of the exception.

EXCMNSTO (TYPE-T, 8 BYTES)
Finish time of the exception.

Note: The performance class exception wait time field, EXWTTIME (103), is a calculation based on subtracting the start time of the exception (EXCMNSTA) from the finish time of the exception (EXCMNSTO).

EXCMNTNO (TYPE-P, 4 BYTES)
Transaction identification number.

EXCMNTPR (TYPE-C, 4 BYTES)
Transaction priority when monitoring was initialized for the task (low-order byte).

EXCMNLUN (TYPE-C, 4 BYTES)
VTAM logical unit name (if available) of the terminal associated with this transaction. This field is nulls if the task is not associated with a terminal.

EXCMNEXN (TYPE-A, 4 BYTES)
Exception sequence number for this task.

EXCMNRTY (TYPE-C, 8 BYTES)
Exception resource type. The possible values for EXCMNRTY are shown in Table 8 on page 106.

EXCMNRID (TYPE-C, 8 BYTES)
Exception resource identification. The possible values for EXCMNRID are shown in Table 8 on page 106.

EXCMNTYP (TYPE-A, 2 BYTES)
Exception type. This field can be set to one of the following values:

X'0001'
Exception due to a wait (EXCMNWT)

X'0002'
Exception due to a buffer wait (EXCMNBWT)

X'0003'
Exception due to a string wait (EXCMNSWT)

EXCMNTCN (TYPE-C, 8 BYTES)
Transaction class name. This field is null if the transaction is not in a transaction class.

EXCMNSRV (TYPE-C, 8 BYTES)
MVS Workload Manager Service Class name for this transaction. This field is null if the transaction was WLM-classified in another CICS region.

EXCMNRPT (TYPE-C, 8 BYTES)
MVS Workload Manager Report Class name for this transaction. This field is null if the transaction was WLM-classified in another CICS region.

EXCMNNPX (TYPE-C, 20 BYTES)
Fully qualified name by which the originating system is known to the VTAM network. This name is assigned at attach time using either the NETNAME derived from the TCT (when the task is attached to a local terminal), or the NETNAME passed as part of an ISC APPC or IRC attach header. At least three passing bytes (X'00') are present at the right end of the name.

If the originating terminal is a VTAM device across an ISC APPC or IRC link, the NETNAME is the *networkid.LUname*. If the terminal is non-VTAM, the NETNAME is *networkid.generic_applid*

All originating information passed as part of an ISC LUTYPE6.1 attach header has the same format as the non-VTAM terminal originators above.

When the originator is communicating over an external CICS interface (EXCI) session, the name is a concatenation of:

'DFHEXCIU	.	MVS Id	Address space Id (ASID)'
8 bytes	1 byte	4 bytes	4 bytes

derived from the originating system. That is, the name is a 17-byte LU name consisting of:

- An 8-byte eye-catcher set to 'DFHEXCIU'.
- A 1-byte field containing a period (.).
- A 4-byte field containing the MVSID, in characters, under which the client program is running.
- A 4-byte field containing the address space ID (ASID) in which the client program is running. This field contains the 4-character EBCDIC representation of the 2-byte hex address space ID.

EXCMNNSX (TYPE-C, 8 BYTES)
Name by which the unit of work is known within the originating system. This last name is assigned at attach time using either an STCK-derived token (when the task is attached to a local terminal) or the unit of work ID is passed as part of an ISC APPC or IRC attach header.

The first 6 bytes of this field are a binary value derived from the clock of the originating system and wrapping round at intervals of several months. The last

two bytes of this field are for the period count. These may change during the life of the task as a result of syncpoint activity.

Note: When using MRO or ISC, the EXCMNNSX field must be combined with the EXCMNNPX field to uniquely identify a task, because the EXCMNNSX field is unique only to the originating CICS system.

EXCMNTRF (TYPE-C, 8 BYTES)
Transaction flags—a string of 64 bits used for signaling transaction definition and status information:

Byte 0 Transaction facility identification

Bit 0 Transaction facility name = none

Bit 1 Transaction facility name = terminal

Bit 2 Transaction facility name = surrogate

Bit 3 Transaction facility name = destination

Bit 4 Transaction facility name = 3270 bridge

Bits 5–7
Reserved

Byte 1 Transaction identification information

Bit 0 System transaction

Bit 1 Mirror transaction

Bit 2 DPL mirror transaction

Bit 3 ONC RCP alias transaction

Bit 4 WEB alias transaction

Bit 5 3270 bridge transaction

Bit 6 Reserved

Bit 7 CICS BTS Run transaction

Byte 2 MVS Workload Manager information

Bit 0 Workload Manager report

Bit 1 Workload Manager notify, completion = yes

Bit 2 Workload Manager notify

Bits 3–7
Reserved

Byte 3 Transaction definition information

Bit 0 Taskdataloc = below

Bit 1 Taskdatakey = cics

Bit 2 Isolate = no

Bit 3 Dynamic = yes

Bits 4– 7
Reserved

Byte 4 Transaction origin type

Byte 5 Reserved

Byte 6 Reserved

Byte 7 Recovery manager information

Bit 0 Indoubt wait = no

Bit 1 Indoubt action = commit

Bit 2 Recovery manager - UOW resolved with indoubt action

Bit 3 Recovery manager - shunt

Bit 4 Recovery manager - unshunt

Bit 5 Recovery manager - indoubt failure

Bit 6 Recovery manager - resource owner failure

Bit 7 Reserved

Note: Bits 2 through 6 will be reset on a SYNCPOINT request when the MSYNC=YES option is specified.

EXCMNFCN (TYPE-C, 4 BYTES)
Transaction facility name. This field is null if the transaction is not associated with a facility. The transaction facility type (if any) can be identified by using byte 0 of the transaction flags field, EXCMNTRF.

EXCMNCPN (TYPE-C, 8 BYTES)
The name of the currently running program for this user task when the exception condition occurred.

EXCMNBTR (TYPE-C, 4 BYTES)
3270 Bridge transaction identification.

EXCMNURI (TYPE-C, 16 BYTES)
RRMS/MVS unit-of-recovery ID (URID)

EXCMNRIL (TYPE-A, 4 BYTES)
Exception resource ID length.

EXCMNRIX (TYPE-C, 256 BYTES)
Exception resource ID (extended).

The following table shows the value and relationships of the fields EXCMNTYP, EXCMNRTY, and EXCMNRID.

Table 8. Possible values of EXCMNTYP, EXCMNRTY, and EXCMNRID. The relationship between exception type, resource type, and resource identification.

EXCMNTYP Exception type	EXCMNRTY Resource type	EXCMNRID Resource ID	MEANING
EXCMNWT	'CFDTLRSW'	poolname	Wait for CF data tables locking request slot
EXCMNWT	'CFDTPOOL'	poolname	Wait for CF data tables non-locking request slot
EXCMNWT	'STORAGE'	'UDSA'	Wait for UDSA storage
EXCMNWT	'STORAGE'	'EUDSA'	Wait for EUDSA storage
EXCMNWT	'STORAGE'	'CDSA'	Wait for CDSA storage
EXCMNWT	'STORAGE'	'ECDSA'	Wait for ECDSA storage
EXCMNWT	'STORAGE'	'SDSA'	Wait for SDSA storage
EXCMNWT	'STORAGE'	'ESDSA'	Wait for ESDSA storage
EXCMNWT	'STORAGE'	'RDSA'	Wait for RDSA storage

Table 8. Possible values of EXCMNTYP, EXCMNRTY, and EXCMNRID (continued). The relationship between exception type, resource type, and resource identification.

EXCMNTYP Exception type	**EXCMNRTY Resource type**	**EXCMNRID Resource ID**	**MEANING**
EXCMNWT	'STORAGE'	'ERDSA'	Wait for ERDSA storage
EXCMNWT	'TEMPSTOR'	TS Qname	Wait for temporary storage
EXCMNSWT	'FILE'	filename	Wait for string associated with file
EXCMNSWT	'LSRPOOL'	filename	Wait for string associated with LSRPOOL
EXCMNSWT	'TEMPSTOR"	TS Qname	Wait for string associated with DFHTEMP
EXCMNBWT	'LSRPOOL'	LSRPOOL	Wait for buffer associated with LSRPOOL
EXCMNBWT	'TEMPSTOR'	TS Qname	Wait for buffer associated with DFHTEMP

Chapter 7. Tivoli Performance Reporter for OS/390

Tivoli Performance Reporter for OS/390, Version 1 Release 3, previously known as Performance Reporter for MVS, supercedes Service Level Reporter (SLR).

Tivoli Performance Reporter for OS/390 is described in the following sections:

- "Overview."
- "Using Tivoli Performance Reporter for OS/390 to report on CICS performance" on page 111

Overview.

Tivoli Performance Reporter for OS/390 is a reporting system which uses DB2. You can use it to process utilization and throughput statistics written to log data sets by computer systems. You can use it to analyze and store the data into DB2, and present it in a variety of forms. Tivoli Performance Reporter consists of a base product with several optional features that are used in systems management, as shown in Table 9. Tivoli Performance Reporter for OS/390 uses Reporting Dialog/2 as the OS/2® reporting feature.

Table 9. Tivoli Performance Reporter for OS/390 and optional features

CICS Performance	IMS Performance	Network Performance	System Performance	Workstation Performance	AS/400® Performance	Reporting Dialog/2	Accounting
Tivoli Performance Reporter for OS/390 Base							

The Tivoli Performance Reporter for OS/390 base includes:

- Reporting and administration dialogs that use the Interactive System Productivity Facility (ISPF)
- A collector function to read log data, with its own language
- Record mapping (definitions) for all data records used by the features

Each feature provides:

- Instructions (in the collector language) to transfer log data to DATABASE 2 (DB2) tables
- DB2 table definitions
- Reports.

The Tivoli Performance Reporter for OS/390 database can contain data from many sources. For example, data from System Management Facilities (SMF), Resource Measurement Facility (RMF), CICS, and Information Management System (IMS) can be consolidated into a single report. In fact, you can define any non-standard log data to Tivoli Performance Reporter for OS/390 and report on that data together with data coming from the standard sources.

The Tivoli Performance Reporter for OS/390 CICS performance feature provides reports for your use when analyzing the performance of CICS Transaction Server for OS/390, and CICS/ESA, based on data from the CICS monitoring facility (CMF) and, for CICS Transaction Server for OS/390, CICS statistics. These are some of the areas that Tivoli Performance Reporter can report on:

- Response times

- Resource usage
- Processor usage
- Storage usage
- Volumes and throughput
- CICS/DB2 activity
- Exceptions and incidents
- Data from connected regions, using the unit of work as key
- SYSEVENT data
- CICS availability
- CICS resource availability

The Tivoli Performance Reporter for OS/390 CICS performance feature collects only the data required to meet CICS users' needs. You can combine that data with more data (called *environment data*), and present it in a variety of reports. Tivoli Performance Reporter for OS/390 provides an administration dialog for maintaining environment data. Figure 12 illustrates how data is organized for presentation in Tivoli Performance Reporter for OS/390 reports.

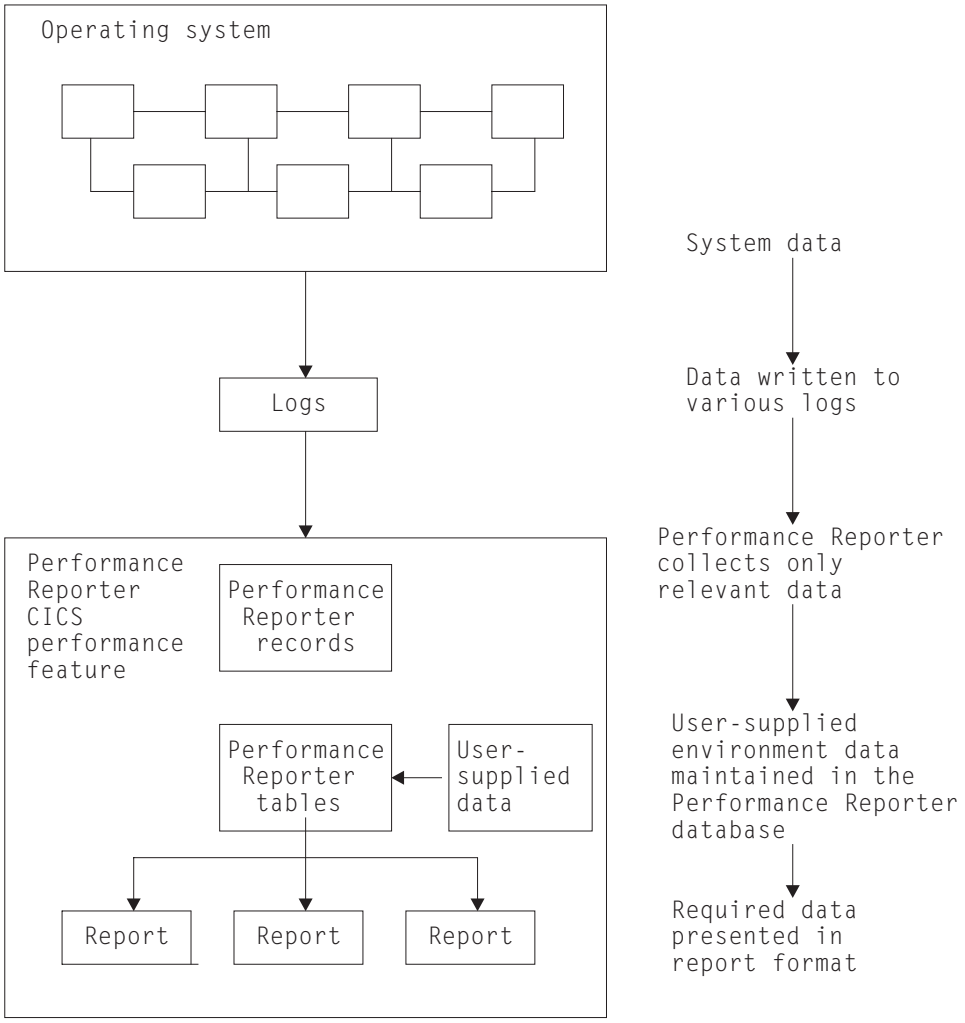


Figure 12. Organizing and presenting system performance data

The Tivoli Performance Reporter for OS/390 CICS performance feature processes these records:

CMF

CICS Transaction Server performance
CICS/ESA performance
CICS/ESA exceptions
CICS/MVS accounting, performance, and exceptions

Statistics

CICS Transaction Server statistics

Using Tivoli Performance Reporter for OS/390 to report on CICS performance

To understand performance data, you must first understand the work CICS performs at your installation. Analyze the work by its basic building blocks: transactions. Group the transactions into categories of similar resource or user requirements and describe each category's characteristics. Understand the work that CICS performs for each transaction and the volume of transactions expected during any given period. Tivoli Performance Reporter for OS/390 can show you various types of data for the transactions processed by CICS.

A service-level agreement for a CICS user group defines commitments in several areas of quantifiable CICS-related resources and services. CICS service commitments can belong to one of these areas:

- Response times
- Transaction rates
- Exceptions and incidents
- Availability.

The following sections describe certain issues and concerns associated with systems management and how you can use the Tivoli Performance Reporter for OS/390 CICS performance feature.

Monitoring response time

Use the Tivoli Performance Reporter for OS/390 CICS response-time reports to see the CICS application internal response times, whose elements are shown in Figure 13.

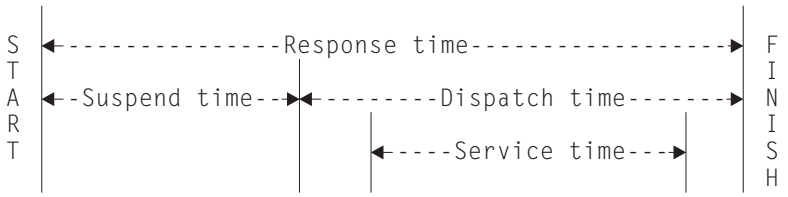


Figure 13. CICS internal response-time elements

As described in *Performance Reporter Network Performance Feature Reports*, the Network Performance feature generates reports that show the total, end-to-end average response time (operator transit time) for VTAM applications (for example, a CICS region) by logical unit. The operator transit time consists of the host transit time and the network transit time, which are also shown in the Network Performance feature reports. Using these reports, you can isolate a response-time problem either to the network or to CICS and act on it accordingly. Should the

problem be in CICS, you can use the Tivoli Performance Reporter for OS/390 CICS performance feature reports to identify the application causing the response-time degradation.

Monitoring processor and storage use

Poor response time usually indicates inefficient use of either the processor or storage (or both). Tivoli Performance Reporter-supplied reports can help you isolate a resource as the cause of a CICS performance problem.

If both the Tivoli Performance Reporter for OS/390 CICS performance feature's statistics component and the Performance Reporter System Performance feature's MVS component are installed and active, these reports are available for analyzing transaction rates and processor use by CICS region:

- The CICS Transaction Processor Utilization, Monthly report shows monthly averages for the dates you specify.
- The CICS Transaction Processor Utilization, Daily report shows daily averages for the dates you specify.

Tivoli Performance Reporter for OS/390 produces several reports that can help analyze storage usage. For example, the CICS Dynamic Storage (DSA) Usage report, shows pagepool usage.

```
                    CICS Dynamic Storage (DSA) Usage
                  MVS ID ='IPO2'      CICS ID ='CSRT5'
                   Date:  '1998-09-21' to '1998-09-22'


                                        Free      Free      Largest
Pagepool          DSA      Cushion     storage   storage     free
  name          (bytes)    (bytes)     (bytes)   (pct)       area     Getmains  Freemains
--------  ---------  --------  ---------  -------  ---------  --------  ---------
CDSA        1048576     65536     802816       76     765952      3695       3620
ECDSA       8388608    262144    7667712       91    7667712      8946       7252
ERDSA       3145728    262144    1302528       41    1290240       204          3
EUDSA       8388608    262144    8388608      100    8388608         1          1
UDSA        4194304     65536    4186112       99    4182016         6          4

                  Tivoli Performance Reporter Report: CICS809
```

Figure 14. CICS Dynamic storage (DSA) usage report

Monitoring volumes and throughput

Because CICS Transaction Server for OS/390 uses an MVS subtask to page and because an MVS page-in causes an MVS task to halt execution, the number of page-ins is a performance concern. Page-outs are not a concern because page-outs are scheduled to occur during lulls in CICS processing. If you suspect that a performance problem is related to excessive paging, you can use Tivoli Performance Reporter for OS/390 to report on page-ins, using RMF data.

The best indicator of a transaction's performance is its response. For each transaction ID, the CICS transaction performance detail report (in Figure 15 on page 113) shows the total transaction count and the average response time.

CICS Transaction Performance, Detail
MVS ID ='IPO2' CICS ID ='CFGTV1 '
Date: '1998-09-19' to '1998-09-20'

Tran ID	Tran count	Avg resp time (sec)	Avg CPU time (sec)	Prog load reqs (avg)	Prog loads (avg)	FC calls (avg)	Excep- tions	Program storage bytes (max)	Getmains < 16 MB (avg)	Getmains > 16 MB (avg)
QUIT	7916	0.085	0.017	0	0	18	0	74344	22	0
CRTE	1760	4.847	0.004	0	0	0	0	210176	1	0
AP00	1750	0.184	0.036	0	0	8	0	309800	66	0
PM94	1369	0.086	0.012	0	0	6	0	130096	24	0
VCS1	737	0.073	0.008	2	0	7	0	81200	14	0
PM80	666	1.053	0.155	1	0	62	0	104568	583	0
CESN	618	8.800	0.001	0	0	0	0	41608	0	0
SU01	487	0.441	0.062	4	0	126	0	177536	38	0
...										
GC11	1	0.341	0.014	1	0	2	0	37048	10	0
DM08	1	0.028	0.002	0	0	0	0	5040	3	0
	20359							309800		

Tivoli Performance Reporter Report: CICS101

Figure 15. CICS transaction performance, detail report

Use this report to start verifying that you are meeting service-level objectives. First, verify that the values for average response time are acceptable. Then check that the transaction rates do not exceed agreed-to limits. If a transaction is not receiving the appropriate level of service, you must determine the cause of the delay.

Combining CICS and DB2 performance data

For each CICS task, CICS generates an LU6.2 unit-of-work ID. DB2 also creates an LU6.2 unit-of-work ID. Figure 16 shows how DB2 data can be correlated with CICS performance data using the DB2 token (QWHCTOKN) to identify the task.

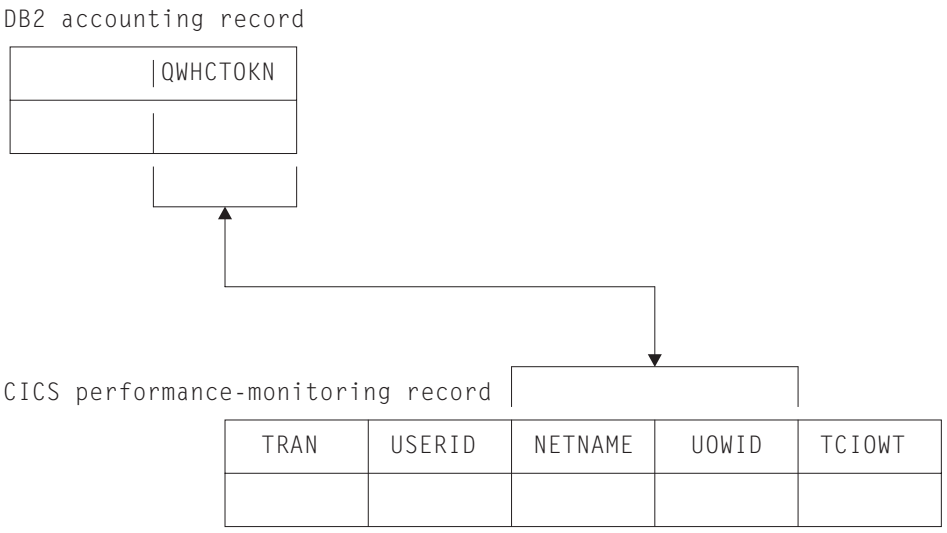


Figure 16. Correlating a CICS performance-monitoring record with a DB2 accounting record

If you match the NETNAME and UOWID fields in a CICS record to the DB2 token, you can create reports that show the DB2 activity caused by a CICS transaction.

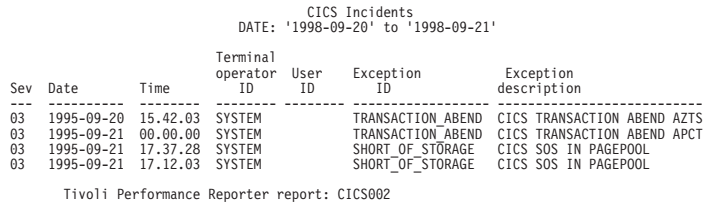
Monitoring exception and incident data

An *exception* is an event that you should monitor. An exception appears in a report only if it has occurred; reports do not show null counts. A single exception need not be a cause for alarm. An incident is defined as an exception with severity 1, 2, or 3.

The Tivoli Performance Reporter for OS/390 CICS performance feature creates exception records for these incidents and exceptions:

- Wait for storage
- Wait for main temporary storage
- Wait for a file string
- Wait for a file buffer
- Wait for an auxiliary temporary storage string
- Wait for an auxiliary temporary storage buffer
- Transaction ABEND
- System ABEND
- Storage violations
- Short-of-storage conditions
- VTAM request rejections
- I/O errors on auxiliary temporary storage
- I/O errors on the intrapartition transient data set
- Autoinstall errors
- MXT reached
- DTB overflow
- Link errors for IRC and ISC
- Log stream buffer-full conditions
- CREAD and CWRITE fails (data space problems)
- Local shared resource (LSR) pool (string waits (from A08BKTSW)
- Waits for a buffer in the LSR pool (from A09TBW)
- Errors writing to SMF
- No space on transient-data data set (from A11ANOSP)
- Waits for a transient-data string (from A11STNWT)
- Waits for a transient-data buffer (from A11ATNWT)
- Transaction restarts (from A02ATRCT)
- Maximum number of tasks in a class reached (CMXT) (from A15MXTM)
- Transmission errors (from A06TETE or AUSTETE).

Figure 17 shows an example of an incidents report.



CICS Incidents
DATE: '1998-09-20' to '1998-09-21'

Sev	Date	Time	Terminal operator ID	User ID	Exception ID	Exception description
03	1995-09-20	15.42.03	SYSTEM		TRANSACTION_ABEND	CICS TRANSACTION ABEND AZTS
03	1995-09-21	00.00.00	SYSTEM		TRANSACTION_ABEND	CICS TRANSACTION ABEND APCT
03	1995-09-21	17.37.28	SYSTEM		SHORT_OF_STORAGE	CICS SOS IN PAGEPOOL
03	1995-09-21	17.12.03	SYSTEM		SHORT_OF_STORAGE	CICS SOS IN PAGEPOOL

Tivoli Performance Reporter report: CICS002

Figure 17. Example of a Tivoli Performance Reporter CICS incidents report

Tivoli Performance Reporter for OS/390 can pass the exceptions to an Information/Management system.

Unit-of-work reporting

In a CICS multiple region operation (MRO) or intersystem communication (ISC) environment, you can trace a transaction as it migrates from one region (or processor complex) to another and back. The data lets you determine the total resource requirements of the combined transaction as a unit of work, without having to separately analyze the component transactions in each region. The ability to combine the component transactions of an MRO or ISC series makes possible precise resource accounting and chargeback, and capacity and performance analysis.

The CICS UOW Response Times report in Figure 18 shows an example of how Tivoli Performance Reporter for OS/390 presents CICS unit- of-work response times.

```
                CICS UOW Response Times
              Time: '09.59.00' to '10.00.00'
                     Date: 1998 09-20


Adjusted
  UOW                                    UOW   Response
 start    Tran  CICS      Program        tran     time
  time     ID    ID        name          count   (sec)
--------  ----  --------  --------       -----  --------
09.59.25  OP22  CICSPROD  DFHAPRT            2     0.436
          OP22  CICSPRDC  OEPCPI22

09.59.26  AP63  CICSPRDE  APPM00             2     0.045
          AP63  CICSPROD  DFHAPRT

09.59.26  ARUS  CICSPROD  DFHAPRT            3     0.158
          CSM5  CICSPRDB  DFHMIR
          ARUS  CICSPRDC  AR49000

09.59.27  CSM5  CICSPRDB  DFHMIR             4     0.639
          CSM5  CICSPRDB  DFHMIR
          MQ01  CICSPROD  DFHAPRT
          MQ01  CICSPRDD  CMQ001

 ...

    Tivoli Performance Reporter report: CICS902
```

Figure 18. Tivoli Performance Reporter for OS/390 CICS UOW response times report

Monitoring availability

Users of CICS applications depend on the availability of several types of resources:

- Central site hardware and the operating system environment in which the CICS region runs
- Network hardware, such as communication controllers, teleprocessing lines, and terminals through which users access the CICS region
- CICS region
- Application programs and data. Application programs can be distributed among several CICS regions.

In some cases, an application depends on the availability of many resources of the same and of different types, so reporting on availability requires a complex analysis of data from different sources. Tivoli Performance Reporter for OS/390 can help you, because all the data is in one database.

Monitoring SYSEVENT data

If the SYSEVENT option is used, CICS records at the end of each transaction:

- Transaction ID
- Associated terminal ID
- Elapsed time

This is useful when you require only transaction statistics, rather than the detailed information that CMF produces. In many cases, it may be sufficient to process only this data, since RMF records it as part of its SMF type-72 record. Analysis (and even recording) of SMF records from CMF can then be reserved for those circumstances when the detailed data is needed. Use the Tivoli Performance Reporter System Performance feature (MVS performance component) to report on this data.

When running under goal mode in MVS 5.1.0 and later, CICS performance can be reported in workload groups, service classes, and periods. These are a few examples of Tivoli Performance Reporter reports for CICS in this environment. Figure 20 on page 116 shows how service classes were served by other service

classes. This report is available only when the MVS system is running in goal mode.

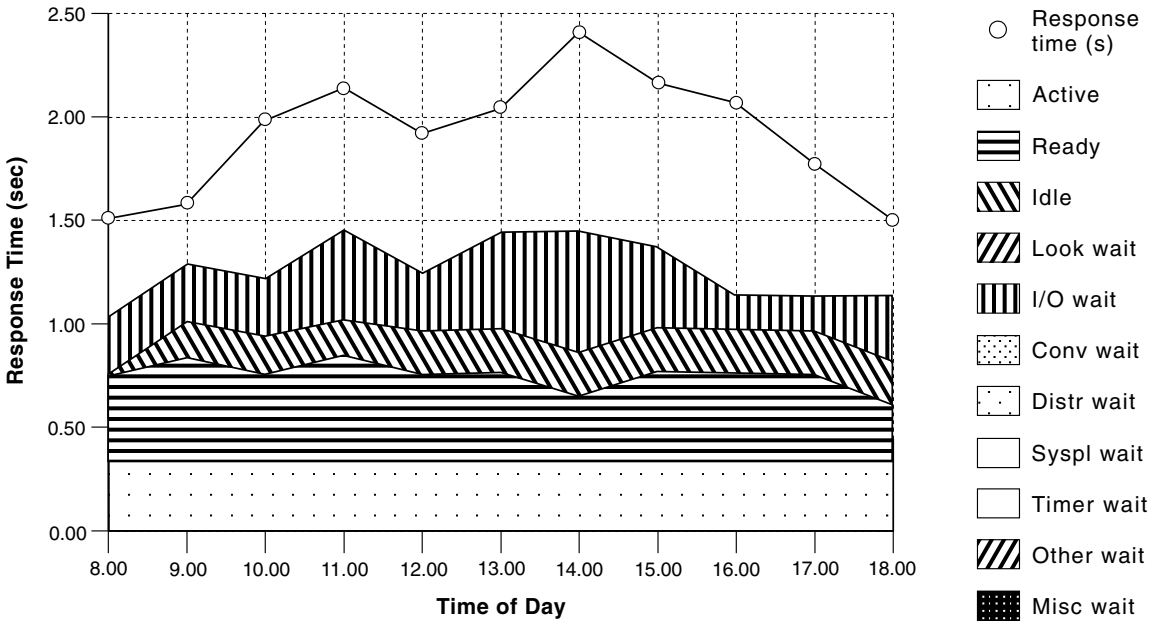


Figure 19. Example of an MVSPM response time breakdown, hourly trend report

```
                    MVSPM Served Service Classes, Overview
                    Sysplex: 'SYSPLEX1' System: MVS_SYSTEM_ID
                    Date: '1998-09-22' Period: 'PRIME'

Workload Service  Served    No of times         No of    No of times
 group    class   class       served            tx's    served per tx
-------- -------- -------- ------------  ------------  --------------
CICS     CICSREGS CICS-1          15227           664            22.9
                  CICS-2           6405           215            29.8
                  CICS-3          24992          1251            20.0
                  CICS-4          87155          1501            58.1
                  CICSTRX         67769          9314             7.3

                Tivoli Performance Reporter report: MVSPM79
```

Figure 20. Example of an MVSPM served service classes overview report

```
                                MVSPM Response Time Breakdown, Overview
                                Sysplex: 'SYSPLEX1' Subsystem: SUBSYSTEM
                                Date: '1995-09-22' Period: 'PRIME'

         Service          MVS  Total Activ Ready Idle   Lock  I/O   Conv Distr Local  Netw Syspl Timer Other  Misc
Workload  class           sys  state state state state  wait  wait  wait wait  wait   wait wait  wait  wait   wait
  group   /Period    Ph  ID     (%)   (%)   (%)   (%)   (%)   (%)   (%)   (%)   (%)   (%)   (%)   (%)   (%)   (%)
-------- ---------- --- ---- ----- ----- ----- ----- ----- ----- ----- -- ----- ----- ----- ----- ----- ----- ----- -----
CICS     CICS-1  /1 BTE CA0    6.6   0.0   0.0   0.0   0.0   0.0   6.5   0.0   0.0   0.0   0.0   0.0   0.0   0.0
                        C80   29.4   0.0   0.0   0.0   0.0   0.0  14.7   0.0   0.0   0.0   0.0   0.0  14.6   0.0
                        C90    3.8   0.4   1.3   1.5   0.0   0.2   0.5   0.0   0.0   0.0   0.0   0.0   0.0   0.0
                           ----- ----- ----- ----- ----- ----- -----  --- ----- ----- ----- ----- ----- ----- -----
                           *  13.3   0.1   0.5   0.5   0.0   0.1   7.2   0.0   0.0   0.0   0.0   0.0   4.9   0.0


                    /1 EXE CA0   16.0   0.1   0.2   0.1   0.0  15.5   0.0   0.0   0.0   0.0   0.0   0.0   0.1   0.0
                        C80   14.9   0.1   0.1   0.1   0.0   3.7   0.0   0.0   0.0   0.0   0.0   0.0  11.0   0.0
                        C90   14.0   1.6   4.5   4.8   0.0   3.2   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0
                           ----- ----- ----- ----- ----- ----- -----  --- ----- ----- ----- ----- ----- ----- -----
                           *  14.9   0.6   1.6   1.7   0.0   7.4   0.0   0.0   0.0   0.0   0.0   0.0   3.7   0.0


IMS      IMS-1   /1 EXE CA0   20.7   0.4   0.7   0.0   0.0   0.0  19.6   0.0   0.0   0.0   0.0   0.0   0.0   0.0
                        C80    1.1   0.2   0.1   0.7   0.0   0.1   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0
                        C90   22.2   5.3  11.9   1.2   0.0   0.2   3.6   0.0   0.0   0.0   0.0   0.0   0.0   0.0
                           ----- ----- ----- ----- ----- ----- -----  --- ----- ----- ----- ----- ----- ----- -----
                           *  14.7   2.0   4.2   0.6   0.0   0.1   7.8   0.0   0.0   0.0   0.0   0.0   0.0   0.0

                Tivoli Performance Reporter report: MVSPM73
```

Figure 21. Example of an MVSPM response time breakdown overview report

Figure 21 shows how much the various transaction states contribute to the average response time. This report is available when the MVS system is running in goal mode and when the subsystem is CICS or IMS.

Figure 19 on page 116 shows the average transaction response time trend and how the various transaction states contribute to it. (The sum of the different states adds up to the average execution time. The difference between the response time and the execution time is mainly made up of switch time, for example, the time the transactions spend being routed to another region for processing). This report is available when the MVS system is running in goal mode and when the subsystem is CICS or IMS.

Chapter 8. Managing Workloads

Workload management in a sysplex is provided by:
- MVS workload manager: see "MVS workload manager"
- CICSPlex SM workload management: see "CICSPlex SM workload management" on page 130

MVS workload manager

This section discusses aspects of the MVS workload manager under the following headings:

MVS/ESA™ 5.1 and later includes the MVS workload manager, which provides automatic, dynamic, balancing of system resources (central processors and storage) across a sysplex by:

- Adopting a goal-oriented approach
- Gathering real-time data from the subsystems that reflect performance at an individual task level
- Monitoring MVS- and subsystem-level delays and waits that are contributing to overall task execution times
- Dynamically managing the sysplex's resources, using the performance goals, and the real-time performance and delay data, as inputs to system resource management algorithms.

This is particularly significant in a sysplex environment, but is also of value to subsystems running in a single MVS image.

CICS provides dynamic routing and distributed routing user-replaceable programs to handle the routing of the work requests to the selected target region. For details, see the *CICS Customization Guide* and *CICS Intercommunication Guide*.

To help you migrate to goal-oriented workload management, you can run any MVS image in a sysplex in *compatibility mode*, using the performance management tuning methods of releases of MVS before MVS/ESA 5.1.

Notes:

1. If you do not want to use the MVS workload management facility, you should review your MVS performance definitions to ensure that they are still appropriate for CICS Transaction Server for OS/390 Release 3. To do this,

review parameters in the IEAICS and IEAIPS members of the MVS PARMLIB library. For more information about these MVS performance definitions, see the *OS/390 MVS Initialization and Tuning Guide*.

2. If you use CICSPlex SM to control dynamic routing in a CICSplex or BTS-plex, you can base its actions on the CICS response time goals of the CICS transactions as defined to the MVS workload manager. See "Using CICSPlex SM workload management" on page 131. For full details, see the *CICSPlex SM Managing Workloads* manual.

Benefits of using MVS Workload Manager

The benefits of using MVS workload manager are:

- Improved performance through MVS resource management

  The improvement is likely to depend on many factors, for example:
  - System hardware configuration
  - The way the system is partitioned
  - Whether CICS subsystems are single or multi-region
  - The spread of types of applications or tasks performed, and the diversity of their profile of operation
  - The extent to which the sysplex workload changes dynamically.
- Improved efficiency of typical MVS sysplexes
  - Improved overall capacity
  - Increased work throughput.
- Simplified MVS tuning

  Generally, systems whose operating signature makes attaining or maintaining optimal tuning difficult or time consuming to achieve by current means will tend to obtain the greater benefit.

The main benefit is that you no longer have to continually monitor and tune CICS to achieve optimum performance. You can set your workload objectives in the service definition and let the workload component of MVS manage the resources and the workload to achieve your objectives.

The MVS workload manager produces performance reports that you can use to establish reasonable performance goals and for capacity planning.

MVS workload management terms

The following terms are used in the description of MVS workload management:

classification rules
: The rules workload management and subsystems use to assign a service class and, optionally, a reporting class to a work request (transaction). A classification rule consists of one or more work qualifiers. See "Defining classification rules" on page 125.

compatibility mode
: A workload management mode for an MVS image in a sysplex using the pre-workload management MVS performance tuning definitions from the IEAICSxx and IEAIPSxx members of the SYS1.PARMLIB library.

goal mode
: A workload management mode for an MVS image in a sysplex using an

MVS workload management service definition to automatically and dynamically balance its system resources according to the active service policy for the sysplex.

report class
Work for which reporting information is collected separately. For example, you can have a report class for information combining two different service classes, or a report class for information on a single transaction.

service class
A subset of a workload having the same service goals or performance objectives, resource requirements, or availability requirements. For workload management, you assign a service goal to a service class. See "Defining service classes" on page 124.

service definition
An explicit definition of all the workloads and processing capacity in a sysplex. A service definition includes service policies, workloads, service classes, resource groups, and classification rules. See "Setting up service definitions" on page 124.

service policy
A set of performance goals for all MVS images using MVS workload manager in a sysplex. There can be only one active service policy for a sysplex, and all subsystems in goal mode within that sysplex process towards that policy. However, you can create several service policies, and switch between them to cater for the different needs of different processing periods.

workload
Work to be tracked, managed and reported as a unit. Also, a group of service classes.

workload management mode
The mode in which workload management manages system resources in an MVS image within a sysplex. The mode can be either compatibility mode or goal mode.

Requirements for MVS workload management

To use MVS workload management you need the following software:

- MVS/ESA System Product (MVS/ESA SP) - JES2 Version 5 Release 1 or a later, upward-compatible, release
- MVS/ESA System Product (MVS/ESA SP) - JES3 Version 5 Release 1 or a later, upward-compatible, release

For MVS workload manager operation across the CICS task-related user exit interface to other subsystems, such as DB2 and DBCTL, you need the appropriate releases of these products.

For more information about requirements for MVS workload management see the following manuals: *MVS Planning: Workload Management*, and *MVS Planning: Sysplex Manager*.

Resource usage

The CICS function for MVS workload management incurs negligible impact on CICS storage.

Span of workload manager operation

MVS workload manager operates across a sysplex. You can run each MVS image in the sysplex in either goal mode or compatibility mode. However, there can be only one active service policy for all MVS images running in goal mode in a sysplex.

All CICS regions (and other MVS subsystems) running on an MVS image with MVS workload manager are subject to the effects of workload management.

If the CICS workload involves non-CICS resource managers, such as DB2 and DBCTL, CICS can pass information through the resource manager interface (RMI[1]) to enable MVS workload manager to relate the part of the workload within the non-CICS resource managers to the part of the workload within CICS.

CICS does not pass information across ISC links to relate the parts of the task execution thread on either side of the ISC link. If you use tasks that communicate across ISC links, you must define separate performance goals, and service classes, for the parts of the task execution thread on each side of the ISC link. These rules apply to ISC links that are:

- Within the same MVS image (so called "intrahost ISC")
- Between MVS images in the same sysplex (perhaps for compatibility reasons)
- Between MVS images in different sysplexes.

If you use tasks that communicate across ISC links between two sysplexes, the separate performance goals are defined in the active service policy for each sysplex.

Defining performance goals

You can define performance goals, such as internal response times, for CICS (and other MVS subsystems that comprise your workload). As an alternative to defining your own goals, you can use "discretionary goals"—the workload manager decides how best to run work for which this type of goal is specified. You can define goals for:

- Individual CICS regions
- Groups of transactions running under CICS
- Individual transactions running under CICS
- Transactions associated with individual userids
- Transactions associated with individual LU names.

Workload management also collects performance and delay data, which can be used by reporting and monitoring products, such as the Resource Measurement Facility (RMF), the TIVOLI Performance Reporter for OS/390, or vendor products.

The service level administrator defines your installation's performance goals, and monitoring data, based on business needs and current performance. The complete definition of workloads and performance goals is called a *service definition*. You may already have this kind of information in a service level agreement (SLA).

Determining CICS response times before defining goals

Before you set goals for CICS work, you can determine CICS current response times by running CICS in compatibility mode with an arbitrary goal. For this purpose, use the SRVCLASS parameter, provided by MVS 5.1 in the installation

1. The CICS interface modules that handle the communication between a task-related user exit and the resource manager are usually referred to as the resource manager interface (RMI) or the task-related user exit (TRUE) interface.

control specification (ICS). This parameter lets you associate a service class with a report performance group, to be run in compatibility mode. You would then:

1. Define a service policy, with a default service class, or classes, for your CICS work, and specify an arbitrary response time goal (say 3 seconds)
2. Define classification rules for the service class or classes (see "Defining classification rules" on page 125)
3. Install the service definition
4. Activate the service policy in compatibility mode.

The average response time for work within the service classes is reported under the report performance group in the RMF Monitor I workload activity report.

This information helps you to set realistic goals for running your CICS work when you switch to goal mode. The reporting data produced by RMF reports:

- Is organized by service class
- Contains reasons for any delays that affect the response time for the service class (for example, because of the actions of a resource manager or an I/O subsystem).

From the reported information, you may be able to determine configuration changes to improve performance.

Example of using SRVCLASS parameter of IEAICSxx

To obtain CICS response time information while in compatibility mode, you can set up the following:

- In your service definition, set up the following:
  - A test policy, comprising the following:

    ```
    Service Policy Name . . . : CICSTEST
    Description . . . . . . . : Migration (compatibility) mode
    ```

  - A workload definition, in which to define the required service class:

    ```
    Workload Name  . . . . . . . : CICSALL
    Description  . . . . . . . . . CICSTEST migration workload
    ```

  - A service class for all CICS transactions:

    ```
    Service Class Name . . . . . : CICSALL
    Description  . . . . . . . . . All CICS transactions
    Workload Name  . . . . . . . . CICSALL
            ---Period---  ---------------------Goal---------------------
    Action  #  Duration   Imp.  Description
      __    1             1    Average response time of 00:00:03.000
    ```

    Note: It does not matter what goal you specify, since it is not used in compatibility mode, but it cannot be discretionary.

  - Specify the name of the service class under the classification rules for the CICS subsystem:

    ```
    Subsystem Type . . . . . . : CICS
    Default Service Class  . . : CICSALL
    ```

- In your ICS member in SYS1.PARMLIB (IEAICS*xx*), specify:

  ```
  SUBSYS=CICS,
  SRVCLASS=CICSALL,RPGN=100
  ```

- Install the workload definition in the coupling facility.
- Activate the test service policy, either by using options provided by the WLM ISPF application, or by issuing the following MVS command:

  ```
  VARY WLM,POLICY=CICSTEST
  ```

You receive response time information about CICS transactions in the RMF Monitor I Workload Activity Report under report performance group 100. For more information about defining performance goals and the use of SRVCLASS, see the *MVS Planning: Workload Management* manual.

Setting up service definitions

You define one service definition for each sysplex. A service definition consists of:

Service policies
See "Defining service policies"

Workloads
See "Defining workloads"

Service classes
See "Defining service classes"

Classification rules
See "Defining classification rules" on page 125

You should record the details of your planned service definition on worksheets, as described in the *MVS Planning: Workload Management* manual. MVS 5.1 provides an ISPF panel-based application for setting up and adjusting the service definition.

Defining service policies

You can have one or more service policies, which are a named set of performance goals meant to cover a certain operating period.

If you have varying performance goals, you can define several service policies.

You can activate only one service policy at a time for the whole sysplex, and, when appropriate, switch to another policy.

Defining workloads

A workload comprises units of work that share some common characteristics that makes it meaningful for an installation to manage or monitor as a group. For example, all CICS work, or all CICS order entry work, or all CICS development work.

A workload is made up of one or more service classes.

Defining service classes

Service classes are categories of work, within a workload, to which you can assign performance goals. You can create service classes for groups of work with similar:

- Performance goals

  You can assign the following performance goals to the service classes:

  Response time
  You can define an average response time (the amount of time required to complete the work) or a response time with percentile (a percentage of work to be completed in the specified amount of time).

  Discretionary
  You can specify that the goal is discretionary for any work for which you do not have specific goals.

Velocity
For work not related to transactions, such as batch jobs and started tasks. For CICS regions started as started tasks, a velocity goal applies only during start-up.

Notes:

1. For service classes for CICS transactions, you cannot define velocity performance goals, discretionary goals, or multiple performance periods.
2. For service classes for CICS regions, you cannot define multiple performance periods.

- Business importance to the installation

  You can assign an importance to a service class, so that one service class goal is recognized as more important than other service class goals. There are five levels of importance, numbered, from highest to lowest, 1 to 5.

You can also create service classes for started tasks and JES, and can assign resource groups to those service classes. You can use such service classes to manage the workload associated with CICS as it starts up, but before CICS transaction-related work begins. (Note that when you define CICS in this way, the address space name is specified as TN, for the task or JES "transaction" name.)

There is a default service class, called SYSOTHER. It is used for CICS transactions for which MVS workload management cannot find a matching service class in the classification rules—for example, if the couple data set becomes unavailable.

Defining classification rules

Classification rules determine how to associate incoming work with a service class. Optionally, the classification rules can assign incoming work to a report class, for grouping report data.

There is one set of classification rules for each service definition. The classification rules apply to every service policy in the service definition; so there is one set of rules for the sysplex.

You should use classification rules for every service class defined in your service definition.

Classification rules categorize work into service classes and, optionally, report classes, based on work qualifiers. You set up classification rules for each MVS subsystem type that uses workload management. The work qualifiers that CICS can use (and which identify CICS work requests to workload manager) are:

LU LU name
LUG LU name group
SI Subsystem instance (VTAM applid)
SIG Subsystem instance group
TN Transaction identifier
TNG Transaction identifier group
UI Userid
UIG Userid group.

Notes:

1. You should consider defining workloads for terminal-owning regions only. Work requests do not normally originate in an application-owning region. They (transactions) are normally routed to an application-owning region from a

terminal-owning region, and the work request is classified in the terminal-owning region. In this case, the work is not reclassified in the application-owning region.

If work orginates in the application-owning region it is classified in the application-owning region; normally there would be no terminal.

2. You can use identifier group qualifiers to specify the name of a group of qualifiers; for example, GRPACICS could specify a group of CICS tranids, which you could specify on classification rules by TNG GRPACICS. This is a useful alternative to specifying classification rules for each transaction separately.

You can use classification groups to group disparate work under the same work qualifier—if, for example, you want to assign it to the same service class.

You can set up a hierarchy of classification rules. When CICS receives a transaction, workload manager searches the classification rules for a matching qualifier and its service class or report class. Because a piece of work can have more than one work qualifier associated with it, it may match more than one classification rule. Therefore, the order in which you specify the classification rules determines which service classes are assigned.

Note: You are recommended to keep classification rules simple.

Example of using classification rules: As an example, you might want all CICS work to go into service class CICSB except for the following:

- All work from LU name S218, except the PAYR transaction, is to run in service class CICSA
- Work for the PAYR transaction (payroll application) entered at LU name S218 is to run in service class CICSC.
- All work from terminals other than LU name S218, and whose LU name begins with S2, is to run in service class CICSD.

You could specify this by the following classification rules:

```
Subsystem Type . . . . . CICS

   -------Qualifier-----------             -------Class--------
   Type       Name     Start               Service     Report
                                 DEFAULTS: CICSB       ________
1  LU         S218                        CICSA       ________
2    TN         PAYR                      CICSC       ________
1  LU         S2*                         CICSD       ________
```

Note: In this classification, the PAYR transaction is nested as a sub-rule under the classification rule for LU name S218, indicated by the number 2, and the indentation of the type and name columns.

Consider the effect of these rules on the following work requests:

```
               Request 1    Request 2    Request 3    Request 4

LU name ......  S218         A001         S218         S214
Transaction ..  PAYR         PAYR         DEBT         ANOT
```

- For request 1, the work request for the payroll application runs in service class CICSC. This is because the request is associated with the terminal with LU name

S218, and the TN—PAYR classification rule specifying service class CICSC is nested under the LU—S218 classification rule qualifier.

- For request 2, the work request for the payroll application runs in service class CICSB, because it is ***not*** associated with LU name S218, nor S2*, and there are no other classification rules for the PAYR transaction. Likewise, any work requests associated with LU names that do not start with S2 run in service class CICSB, as there are classification rules for LU names S218 and S2* only.
- For request 3, the work request for the DEBT transaction runs in service class CICSA, because it is associated with LU name S218, and there is no DEBT classification rule nested under the LU—S218 classification rule qualifiers.
- For request 4, the work request for the ANOT transaction runs in service class CICSD, because it is associated with an LU name starting S2, but not S218.

However, if the classification rules were specified as:

```
1 TN        PAYR                    CICSA      ________
1 LU        S218                    CICSA      ________
2   TN      PAYR                    CICSC      ________
1 LU        S2*                     CICSD      ________
```

the PAYR transaction would always run in service class CICSA, even if it were associated with LU name S218.

Guidelines for classifying CICS transactions

For RMF to provide meaningful Workload Activity Report data it is suggested that you use the following guidelines when defining the service classes for CICS transactions. In the same service class:

1. Do not mix CICS-supplied transactions with user transactions
2. Do not mix routed with non-routed transactions
3. Do not mix conversational with pseudo-conversational transactions
4. Do not mix long-running and short-running transactions.

Using a service definition base

To minimize the amount of data you need to enter into the ISPF workload application, you use a *service definition base*. When you set up your service definition, you identify the workloads, the service classes, and their goals, based on your performance objectives. Then you define classification rules. This information makes up the service definition base. The base contains workloads, service classes, resource groups, report classes, and classification rules.

All workloads, service classes, and classification rules defined in a service definition base apply to every policy that you define. You should use classification rules for every service class defined in your service definition. If you do not have any other business requirements to modify a service goal or a resource group from the service definition base, you can run an installation with one policy.

Using MVS workload manager

To use the MVS workload manager facility, you should:

1. Implement workload management on the MVS images that the CICS workload is to run on, as outlined in “Implementing MVS workload management” on page 128.

2. Ensure that CICS performance parameters correspond to the policies defined for MVS workload management, as outlined in "Matching CICS performance parameters to service policies" on page 129.
3. Activate MVS workload manager, as outlined in "Activating CICS support for MVS workload manager" on page 129.

Implementing MVS workload management

The task of implementing MVS workload management is part of the overall task of planning for, and installing, MVS 5.1.

Implementing MVS workload management generally involves the following steps:

1. Establish your workloads.
2. Set your business priorities.
3. Understand your performance objectives.
4. Define critical work.
5. Define performance objectives based on current:
   - Business needs
   - Performance:
     - Reporting and monitoring products
     - Capacity planning tools
     - IEAICS and IEAIPS parameters.
6. Get agreement for your workload performance objectives.
7. Specify a service level agreement or performance objectives.
8. Specify an MVS WLM service definition using the information from step 7.

   Note: It is helpful at this stage to record your service definition in a form that will help you to enter it into the MVS workload manager ISPF application. You are recommended to use the worksheets provided in the MVS publication *Planning: Workload Management*.
9. Install MVS.
10. Set up a sysplex with a single MVS image, and run in workload manager compatibility mode.
11. Upgrade your existing XCF couple data set.
12. Start the MVS workload manager ISPF application, and use it in the following steps.
13. Allocate and format a new couple data set for workload management. (You can do this from the ISPF application.)
14. Define your service definition.
15. Install your service definition on the couple data set for workload management.
16. Activate a service policy.
17. Switch the MVS image into goal mode.
18. Start up a new MVS image in the sysplex. (That is, attach the new MVS image to the couple data set for workload management, and link it to the service policy.)
19. Switch the new MVS image into goal mode.
20. Repeat steps 18 and 19 for each new MVS image in the sysplex.

Notes:

1. CICS Transaction Server for OS/390 support for MVS workload manager is initialized automatically during CICS startup.
2. All CICS regions (and other MVS subsystems) running on an MVS image with MVS workload management are subject to the effects of workload manager.

Matching CICS performance parameters to service policies

You must ensure that the CICS performance parameters are compatible with the workload manager service policies used for the CICS workload.

In general, you should define CICS performance objectives to the MVS workload manager first, and observe the effect on CICS performance. Once the MVS workload manager definitions are working correctly, you can then consider tuning the CICS parameters to further enhance CICS performance. However, you should use CICS performance parameters as little as possible.

Performance attributes that you might consider using are:

- Transaction priority, passed on dynamic transaction routing. (Use prioritization carefully, if at all.) The priority assigned by the CICS dispatcher must be compatible with the task priority defined to MVS workload manager.
- Maximum number of concurrent user tasks for the CICS region.
- Maximum number of concurrent tasks in each transaction class.

Activating CICS support for MVS workload manager

CICS Transaction Server for OS/390 Release 3 support for MVS workload manager is initialized automatically during CICS startup.

Customer-written resource managers and other non-CICS code which is attached to CICS via the RMI must be modified to provide workload manager support, if workload manager is to work correctly for CICS-based tasks which cross the RMI into such areas.

Explanation of the difference between a DFHSTUP transaction report and an RMF workload report

Figure 22 on page 130 shows the significance of the difference between the performance reports created for the region by DFHSTUP, and those generated by the RMF workload activity report for the reporting performance group number (RPGN). In the terminal-owning region (TOR), the WLM reports for transaction ABCD are included in the workload activity reports for the RPGN defined for the service class CICSPROD.region (AOR). In the application-owning region (AOR), notifys for routed transactions are included when reporting EXECution phases on the CICS AOR workload activity report for the RPGN defined for service class CICSPROD. Transaction WLM notifys for mirror transactions are ignored by the MVS WLM when reporting EXECution phases on the CICS FOR workload activity reports for service class CICSPROD.

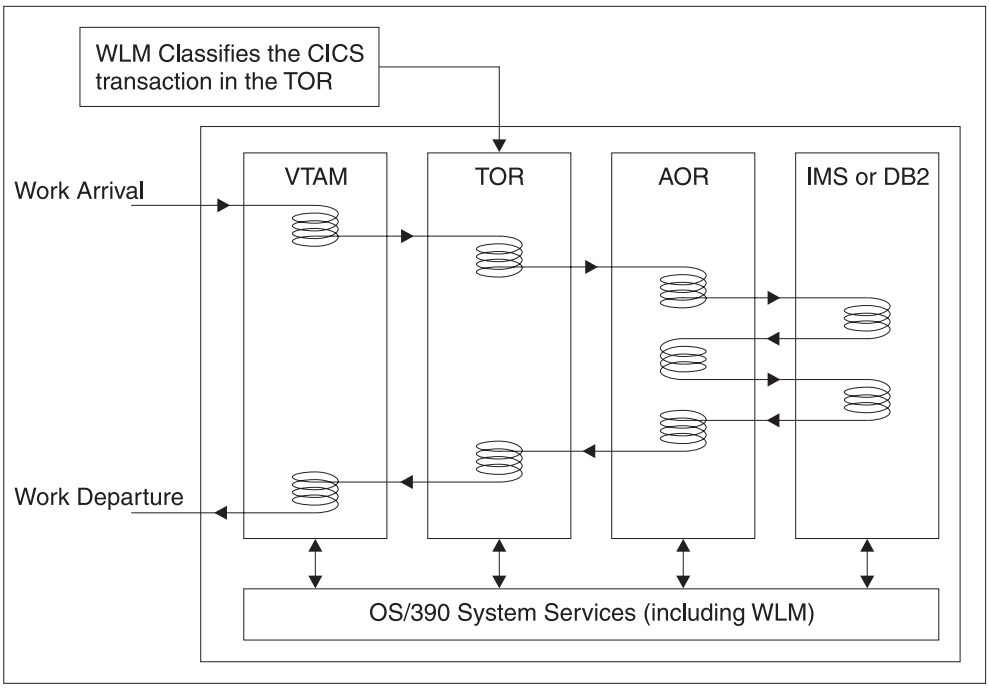


Figure 22. CICS MRO transaction workflow

CICS transaction manager global statistics (see "Program autoinstall" on page 424) include ALL transactions in the interval or summary reports from DFHSTUP, but as shown in Figure 22, the MVS WLM workload activity report includes only the transactions in begin-to-end phase (BTE) and EXECution (BW) phase. For WLM reporting purposes, the EXECution phase applies to only routed transactions in the AOR.

CICSPlex SM workload management

CICSPlex SM workload management directs work requests to a target region that is selected using one of the following:

The queue algorithm
CICSPlex SM routes work requests initiated in the requesting region to the most suitable target region within the designated set of target regions.

The goal algorithm
CICSPlex SM routes work requests to the target region that is best able to meet the goals that have been predefined using MVS workload manager.

The CICSPlex SM dynamic routing program EYU9XLOP is invoked to route work requests to the selected target region. EYU9XLOP supports both workload balancing and workload separation. You define to CICSPlex SM which requesting, routing, and target regions in the CICSplex or BTS-plex can participate in dynamic routing, and any affinities that govern the target regions to which particular work requests must be routed. The output from the Transaction Affinities Utility can be used directly by CICSPlex SM.

For more information about CICSPlex SM, see the *CICSPlex SM Concepts and Planning* manual.

Benefits of using CICSPlex SM workload management

There are no special requirements to be able to use the dynamic transaction routing mechanism, but it offers the user the following:

- A dynamic routing program to make more intelligent routing decisions; for example, based on workload goals.
- CICS can provide improved support for MVS goal-oriented workload management.
- Make it easier to use a global temporary storage owning region in the MVS sysplex environment, which avoids intertransaction affinity that can occur with the use of local temporary storage queues.
- Can be used by CICSPlex SM to provide intelligent routing in a CICSPlex or a BTS-plex that has at least one requesting region linked to multiple target regions.

Using CICSPlex SM workload management

For information on setting up and using CICSPlex SM workload management, see the *CICSPlex SM Concepts and Planning* and the *CICSPlex SM Managing Workloads* manuals.

Chapter 9. Understanding RMF workload manager data

This chapter provides in the following sections an explanation, together with a number of examples, of CICS-related data from an RMF workload activity report.

- "Explanation of terms used in RMF reports"
- "Interpreting the RMF workload activity data" on page 135

RMF provides data for subsystem *work managers* that support workload management. In MVS these are IMS and CICS.

This chapter includes a discussion of some possible data that may be reported for CICS and IMS, and provides some possible explanations for the data. Based on this discussion and the explanations, you may decide to alter your service class definitions. In some cases, there may be some actions that you can take, in which case you can follow the suggestion. In other cases, the explanations are provided only to help you better understand the data. For more information about using RMF, see the *RMF User's Guide*.

Explanation of terms used in RMF reports

It might help to relate some of the terms used in an RMF activity report to the more familiar CICS terms. For example, some of terms in the RMF report can be equated with CEMT INQUIRE TASK terms.

These explanations are given for two main sections of the reports:

- The response time breakdown in percentage section
- The state section, covering switched time.

The response time breakdown in percentage section

The "Response time breakdown in percentage" section of the RMF report contains the following headings:

ACTIVE
: The percentage of response time accounted for by tasks currently executing in the region—tasks shown as *Running* by the CEMT INQUIRE TASK command.

READY
: The percentage of response time accounted for by tasks that are not currently executing but are ready to be dispatched—tasks shown as *Dispatchable* by the CEMT INQUIRE TASK command.

IDLE The percentage of response time accounted for by a number of instances or types of CICS tasks:

- Tasks waiting on a principal facility (for example, conversational tasks waiting for a response from a terminal user)
- The terminal control (TC) task, CSTP, waiting for work
- The interregion controller task, CSNC, waiting for transaction routing requests
- CICS system tasks, such as CSSY or CSNE waiting for work.

A CEMT INQUIRE TASK command would show any of these user tasks as *Suspended*, as are the CICS system tasks.

WAITING FOR
The percentage of response time accounted for by tasks that are not currently executing and are not ready to be dispatched—shown as *Suspended* by the CEMT INQUIRE TASK command.

The WAITING FOR main heading is further broken down into a number of subsidiary headings. Where applicable, for waits other than those described for the IDLE condition described above, CICS interprets the cause of the wait, and records the 'waiting for' reason in the WLM performance block.

The waiting-for terms used in the RMF report equate to the WLM_WAIT_TYPE parameter on the SUSPEND, WAIT_OLDC, WAIT_OLDW, and WAIT_MVS calls used by the dispatcher, and the SUSPEND and WAIT_MVS calls used in the CICS XPI. These are shown as follows (with the CICS WLM_WAIT_TYPE term, where different from RMF, in parenthesis):

Term **Description**

LOCK Waiting on a lock. For example, waiting for:
- A lock on CICS resource
- A record lock on a recoverable VSAM file
- Exclusive control of a record in a BDAM file
- An application resource that has been locked by an EXEC CICS ENQ command.

I/O (IO)
Waiting for an I/O request or I/O related request to complete. For example:
- File control, transient data, temporary storage, or journal I/O.
- Waiting on I/O buffers or VSAM strings.

CONV
Waiting on a conversation between work manager subsystems. This information is further analyzed under the SWITCHED TIME heading.

DIST Not used by CICS.

LOCAL (SESS_LOCALMVS)
Waiting on the establishment of a session with another CICS region in the same MVS image in the sysplex.

SYSPL (SESS_SYSPLEX)
Waiting on establishment of a session with another CICS region in a different MVS image in the sysplex.

REMOT (SESS_NETWORK)
Waiting on the establishment of an ISC session with another CICS region (which may, or may not, be in the same MVS image).

TIMER
Waiting for a timer event or an interval control event to complete. For example, an application has issued an EXEC CICS DELAY or EXEC CICS WAIT EVENT command which has yet to complete.

PROD (OTHER_PRODUCT)
Waiting on another product to complete its function; for example, when the work request has been passed to a DB2 or DBCTL subsystem.

MISC Waiting on a resource that does not fall into any of the other categories.

The state section

The state section covers the time that transactions are "switched" to another CICS region:

SWITCHED TIME
The percentage of response time accounted for by tasks in a TOR that are waiting on a conversation across an intersystem communication link (MRO or ISC). This information provides a further breakdown of the response time shown under the CONV heading.

The SWITCHED TIME heading is further broken down into a number of subsidiary headings, and covers those transactions that are waiting on a conversation. These are explained as follows:

LOCAL
The work request has been switched, across an MRO link, to another CICS region in same MVS image.

SYSPL
The work request has been switched, across an XCF/MRO link, to another CICS region in another MVS image in the sysplex.

REMOT
The work request has been switched, across an ISC link, to another CICS region (which may, or may not, be in the same MVS image).

For more information on the MVS workload manager states and resource names used by CICS Transaction Server for OS/390 Release 3, see the *CICS Problem Determination Guide*.

Interpreting the RMF workload activity data

Figure 24 on page 136 shows an example of the CICS state section of an RMF Monitor I workload activity report. It is based on an example hotel reservations service class.

The text following the figure explains how to interpret the fields.

RMF reporting intervals

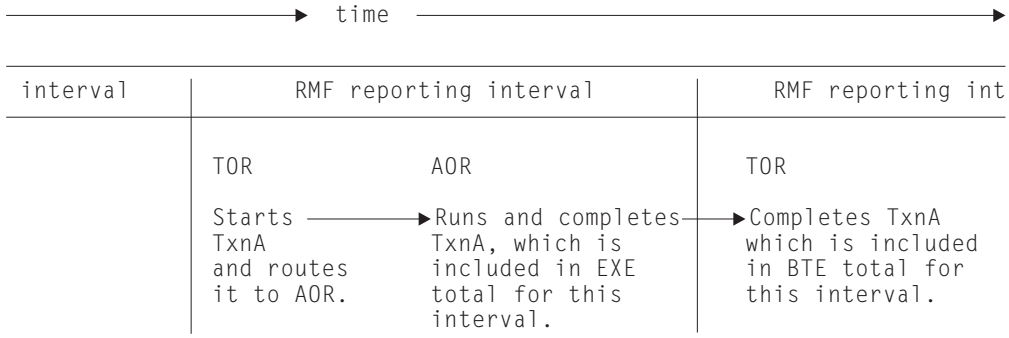


Figure 23. Illustration of snapshot principle for RMF reporting intervals

```
REPORT BY: POLICY=HPTSPOL1  WORKLOAD=PRODWKLD SERVICE CLASS=CICSHR RESOURCE GROUP=*NONE PERIOD=1 IMPORTANCE=HIGH

-TRANSACTIONS--  TRANSACTION TIME   HHH.MM.SS.TTT
AVG        0.00  ACTUAL             000.00.00.114
MPL        0.00  QUEUED             000.00.00.036
ENDED       216  EXECUTION          000.00.00.078
END/SEC    0.24  STANDARD DEVIATION 000.00.00.270
#SWAPS        0
EXECUTD     216

           -------------------------RESPONSE TIME BREAKDOWN IN PERCENTAGE------------------  ----STATE------
SUB   P  TOTAL  ACTIVE  READY IDLE  ------------------------WAITING FOR--------------------  SWITCHED TIME (%)
TYPE                               LOCK I/O  CONV  DIST LOCAL SYSPL  REMOT TIMER  PROD  MISC  LOCAL SYSPL REMOT
CICS BTE  93.4   10.2   0.0   0.0   0.0  0.0  83.3   0.0  0.0   0.0    0.0   0.0    0.0   0.0   83.3  0.0   0.0
CICS EXE  67.0   13.2   7.1   0.0   0.0  0.0   0.0   0.0  0.0   0.0    0.0   0.0   46.7   0.0    0.0  0.0   0.0
```

Figure 24. Hotel Reservations service class

An RMF workload activity report contains “snapshot data” which is data collected over a relatively short interval. The data for a given work request (CICS transaction) in an MRO environment is generally collected for more than one CICS region, which means there can be some apparent inconsistencies between the execution (EXE) phase and the begin to end (BTE) data in the RMF reports. This is caused by the end of a reporting interval occurring at a point when work has completed in one region but not yet completed in an associated region. See Figure 23 on page 135.

For example, an AOR can finish processing transactions, the completion of which are included in the current reporting interval, whilst the TOR may not complete its processing of the same transactions during the same interval.

The fields in this RMF report describe an example CICS hotel reservations service class (CICSHR), explained as follows:

CICS This field indicates that the subsystem work manager is CICS.

BTE This field indicates that the data in the row relates to the *begin-to-end* work phase.

CICS transactions are analyzed over two phases: a begin-to-end (BTE) phase, and an execution (EXE) phase.

The begin-to-end phase usually takes place in the terminal owning region (TOR), which is responsible for starting and ending the transaction.

EXE This field indicates that the data in the row relates to the *execution* work phase. The execution phase can take place in an application owning region (AOR) and a resource-owning region such as an FOR. In our example, the 216 transactions were routed by a TOR to another region for execution, such as an AOR (and possibly an FOR).

ENDED
This field shows that 216 hotel reservation transactions completed.

EXECUTD
This field shows that the AORs completed 216 transactions in the reporting interval.

Note: In our example the two phases show the same number of transactions completed, indicating that during the reporting interval all the transactions routed by the TORs (ENDED) were completed by the AORs (EXECUTD) and also completed by the TORs. This will not normally be the case because of the way data is captured in RMF reporting intervals. See “RMF reporting intervals” on page 135.

ACTUAL
Shown under TRANSACTION TIME, this field shows the average response time as 0.114 seconds, for the 216 transactions completed in the BTE phase.

EXECUTION
Shown under TRANSACTION TIME, this field shows that on average it took 0.078 seconds for the AORs to execute the transactions.

While executing these transactions, CICS records the states the transactions are experiencing. RMF reports the states in the RESPONSE TIME BREAKDOWN IN PERCENTAGE section of the report, with one line for the begin-to-end phase, and another for the execution phase.

The response time analysis for the BTE phase is described as follows:

For BTE
Explanation

TOTAL
The CICS BTE total field shows that the TORs have information covering 93.4% of the ACTUAL response time, the analysis of which is shown in the remainder of the row. This value is the ratio of sampled response times to actual response times. The sampled response times are derived by calculating the elapse times to be the number of active performance blocks (inflight transactions) multiplied by the sample interval time. The actual response times are those reported to RMF by CICS when each transaction ends. The proximity of the total value to 100% and a relatively small standard deviation value are measures of how accurately the sampled data represents the actual system behavior. “Possible explanations” on page 138 shows how these reports can be distorted.

ACTIVE
On average, the work (transactions) was active in the TORs for only about 10.2% of the ACTUAL response time

READY
In this phase, the TORs did not detect that any part of the average response time was accounted for by work that was dispatchable but waiting behind other transactions.

IDLE In this phase, the TORs did not detect that any part of the average response time was accounted for by transactions that were waiting for work.

WAITING FOR
Only one field shows a value in the WAITING FOR section—the CONV value (this is typical for a TOR). It indicates that for about 83.3% of the time, the transactions were waiting on a conversation. This is further explained by the SWITCHED TIME data.

SWITCHED TIME
From the SWITCHED TIME % data you can see the reason for the ‘waiting-on-a-conversation’. This is 83.3 % LOCAL, which indicates that the transactions were routed locally to an AOR on the same MVS image.

Note: In the analysis of the BTE phase, the values do not exactly add up to the TOTAL value because of rounding—in our example, 10.2 + 83.3 = 93.5, against a total shown as 93.4.

The response time analysis for the EXE phase is described as follows:

For EXE

Explanation

TOTAL

The CICS EXE total field shows that the AORs have information covering 67% of the ACTUAL response time.

ACTIVE

On average, the work is active in the AOR for only about 13.2% of the average response time.

READY

On average the work is ready, but waiting behind other tasks in the region, for about 7.1% of the average response time.

PROD On average, 46.7% of the average response time is spent outside the CICS subsystem, waiting for another product to provide some service to these transactions.

You can't tell from this RMF report what the other product is, but the probability is that the transactions are accessing data through a database manager such as Database Control (DBCTL) or DB2.

Example: very large percentages in the response time breakdown

Figure 25 shows an example of a work manager state section for the CICSPROD service class. In the RESPONSE TIME BREAKDOWN IN PERCENTAGE section of the report, both the CICS EXE and the CICS BTE rows show excessively inflated percentages: 78.8K, 183, 1946 and so on.

```
REPORT BY: POLICY=HPTSPOL1 WORKLOAD=PRODWKLD SERVICE CLASS=CICSPROD  RESOURCE GROUP=*NONE  PERIOD=1 IMPORTANCE=HIGH

-TRANSACTIONS--  TRANSACTION TIME   HHH.MM.SS.TTT
AVG        0.00  ACTUAL             000.00.00.111
MPL        0.00  QUEUED             000.00.00.000
ENDED      1648  EXECUTION          000.00.00.123
END/SEC    1.83  STANDARD DEVIATION 000.00.00.351
#SWAPS        0
EXECUTD    1009

            -------------------------------RESPONSE TIME BREAKDOWN IN PERCENTAGE----------------------   ---STATE--------
SUB    P  TOTAL  ACTIVE  READY  IDLE    ------------------------WAITING FOR--------------------------   SWITCHED TIME (%)
TYPE                                    LOCK   I/O  CONV  DIST LOCAL SYSPL REMOT  TIMER   PROD  MISC   LOCAL SYSPL REMOT
CICS  BTE  78.8K   183    265   1946    0.0    0.0   235   0.0   0.0   0.0   0.0   0.0    0.0  76.2K    229   0.0  17.9
CICS  EXE   140    91.8   3.1    0.0    0.0    0.1   0.0   0.0   0.0   0.0   0.0   0.0   45.4   0.0    19.6K  0.0   0.0
```

Figure 25. Response Time percentages greater than 100

Possible explanations

There several possible explanations for the unusual values shown in this sample report:

- Long-running transactions
- Never-ending transactions
- Conversational transactions
- Dissimilar work in service class

Long-running transactions

The RMF report in Figure 24 on page 136 shows both very high response times percentages and a large standard deviation of reported transaction times.

The report shows for the recorded 15 minute interval that 1648 transactions completed in the TOR. These transactions had an actual average response time of

0.111seconds (note that this has a large standard deviation) giving a total of 182.9 seconds running time (0.111 seconds multiplied by 1648 transactions). However, if there are a large number of long running transactions also running, these will be counted in the sampled data but not included in the the actual response time values. If the number of long running transactions is large, the distortion of the Total value will also be very large.

The long running transactions could be either routed or non-routed transactions. Routed transactions are transactions that are routed from a TOR to one or more AORs. Long-running routed transactions could result in many samples of waiting for a conversation (CONV) in the CICS begin-to-end phase, with the AOR's state shown in the execution phase.

Non-routed transactions execute completely in a TOR, and have no execution (CICS EXE) phase data. Non-routed CICS transactions could inflate the ACTIVE or READY data for the CICS BTE phase.

Never-ending transactions

Never-ending transactions differ from long-running transactions in that they persist for the life of a region. For CICS, these could include the IBM reserved transactions such as CSNC and CSSY, or customer defined transactions. Never-ending transactions are reported in a similar way to long-running transactions, as explained above. However, for never-ending CICS transactions, RMF might report large percentages in IDLE, or under TIMER or MISC in the WAITING FOR section.

Conversational transactions

Conversational transactions are considered long-running transactions. CICS marks the state of a conversational transaction as IDLE when the transaction is waiting for terminal input. Terminal input often includes long end-user think time, so you might see very large values in the IDLE state as a percent of response time for completed transactions.

Dissimilar work in the service class

A service class that mixes:

- Customer and IBM transactions,
- Long-running and short-running transactions
- Routed and non-routed transactions
- Conversational and non-conversational transactions

can expect to have RMF reports showing that the total states sampled account for more than the average response time. This can be expected if the service class is the subsystem default service class. The default is defined in the classification rules as the service class to be assigned to all work in a subsystem not otherwise assigned a service class.

Possible actions

The following are some actions you could take for reports of this type:

Group similar work into the same service classes: Make sure your service classes represent groups of similar work. This could require creating additional service classes. For the sake of simplicity, you may have only a small number of service classes for CICS work. If there are transactions for which you want the RMF response time breakdown data, consider including them in their own service class.

Do nothing: For service classes representing dissimilar work such as the subsystem default service class, recognize that the response time breakdown could include long-running or never-ending transactions. Accept that RMF data for such service classes does not make much sense.

Example: response time breakdown data is all zero

Figure 26 shows an example of a work manager state section for the CICSLONG service class. All data shows a 0.0 value.

```
REPORT BY: POLICY=HPTSPOL1  WORKLOAD=PRODWKLD SERVICE CLASS=CICSLONG  RESOURCE GROUP=*NONE  PERIOD=1 IMPORTANCE=HIGH
                                                    CICS Long Running Internal Trxs

-TRANSACTIONS--  TRANSACTION TIME   HHH.MM.SS.TTT
AVG        0.00  ACTUAL             000.00.00.000
MPL        0.00  QUEUED             000.00.00.000
ENDED         0  EXECUTION          000.00.00.000
END/SEC    0.00  STANDARD DEVIATION 000.00.00.000
#SWAPS        0
EXECUTD       0

           -------------------------------RESPONSE TIME BREAKDOWN IN PERCENTAGE---------------   ---------STATE---
SUB    P   TOTAL  ACTIVE  READY  IDLE  ----------------------------WAITING FOR----------------   SWITCHED TIME (%)
TYPE                                    LOCK  I/O  CONV  DIST LOCAL SYSPL REMOT TIMER PROD MISC   LOCAL SYSPL REMOT
CICS  BTE   0.0     0.0    0.0    0.0  0.0   0.0  0.0   0.0   0.0   0.0   0.0   0.0   0.0  0.0    0.0   0.0   0.0
```

Figure 26. Response time breakdown percentages all 0.0

Possible explanations

There are two possible explanations:

1. No transactions completed in the interval
2. RMF did not receive data from all systems in the sysplex.

No transactions completed in the interval

While a long-running or never-ending transaction is being processed, RMF saves the service class state samples to SMF Type 72 records, (subtype 3). But when no transactions have completed, (and average response time is 0), the calculations to apportion these state samples over the response time result in 0%.

RMF did not receive data from all systems in the sysplex.

The RMF post processor may have been given SMF records from only a subset of the systems running in the sysplex. For example, the report may represent only a single MVS image. If that MVS image has no TOR, its AORs receive CICS transactions routed from another MVS image or from outside the sysplex. Since the response time for the transactions is reported by the TOR, there is no transaction response time for the work, nor are there any ended transactions.

Possible actions

The following are some actions you could take for reports of this type:

Do nothing

You may have created this service class especially to prevent the state samples of long running transactions from distorting data for your production work. In this case there is no action to take.

Combine all SMF records for the sysplex

The state data is contained in the SMF records. If you combine the data from an MVS image that doesn't have a TOR with another MVS image that does, the state data from the two MVS images is analyzed together by RMF. This ensures that the response time distribution data is no longer reported as zeros.

Example: execution time greater than response time

Figure 27 shows an example of a work manager state section for the CICSPROD service class. In the example, there are 1731 ENDED transactions yet the EXECUTD field shows that only 1086 have been executed. The response time (ACTUAL field) shows 0.091 seconds as the average of all 1731 transactions, while the AORs can only describe the execution of the 1086 they participated in, giving an execution time of 0.113.

```
REPORT BY: POLICY=HPTSPOL1  WORKLOAD=PRODWKLD  SERVICE CLASS=CICSPROD RESOURCE GROUP=*NONE   PERIOD=1 IMPORTANCE=HIGH
                                                                    CICS Trans not classified singly

 -TRANSACTIONS--  TRANSACTION TIME   HHH.MM.SS.TTT
 AVG        0.00  ACTUAL             000.00.00.091
 MPL        0.00  QUEUED             000.00.00.020
 ENDED      1731  EXECUTION          000.00.00.113
 END/SEC    1.92  STANDARD DEVIATION 000.00.00.092
 #SWAPS        0
 EXECUTD    1086
```

Figure 27. Execution time greater than response time

Possible explanation

The situation illustrated by this example could be explained by the service class containing a mixture of routed and non-routed transactions. In this case, the AORs have recorded states which account for more time than the average response time of all the transactions. The response time breakdown shown by RMF for the execution phase of processing can again show percentages exceeding 100% of the response time.

Possible actions

Define routed and non-routed transactions in different service classes.

Example: large SWITCH LOCAL Time in CICS execution phase

Figure 28 shows a work manager state data section for a CICSPROD service class. The SWITCH LOCAL time in the response time breakdown section shows a value of 6645.

```
REPORT BY: POLICY=HPTSPOL1  WORKLOAD=PRODWKLD SERVICE CLASS=CICSPROD  RESOURCE GROUP=*NONE  PERIOD=1 IMPORTANCE=HIGH

 -TRANSACTIONS--  TRANSACTION TIME   HHH.MM.SS.TTT
 AVG        0.00  ACTUAL             000.00.00.150
 MPL        0.00  QUEUED             000.00.00.039
 ENDED      3599  EXECUTION          000.00.00.134
 END/SEC    4.00  STANDARD DEVIATION 000.00.00.446
 #SWAPS        0
 EXECUTD    2961

          ------------------------------RESPONSE TIME BREAKDOWN IN PERCENTAGE-----------------   ------STATE------
 SUB    P   TOTAL ACTIVE READY  IDLE ---------------------------WAITING FOR----------------------   SWITCHED TIME (%)
 TYPE                              LOCK  I/O  CONV  DIST  LOCAL SYSPL REMOT TIMER  PROD  MISC   LOCAL SYSPL REMOT
 CICS  BTE  26.8K  75.1  98.4    659   0.0   0.3   154  0.0    0.0   0.0   0.0   0.0   0.0   25.8K  149   0.0   7.8
 CICS  EXE   93.7  38.6   5.6    0.0   0.0   0.1   0.0  0.0    0.0   0.0   0.0   0.0   49.4   0.0  6645   0.0   0.0
```

Figure 28. High SWITCH time in a CICS execution environment

Possible explanations

This situation can be explained by instances of distributed transaction processing

If, while executing a transaction, an AOR needs to function ship a request to another region (for example, to a file-owning or queue-owning region), the

execution time reported in the RMF report for the AOR (the CICS EXE field) includes the time spent in that other region.

However, if a program initiates distributed transaction processing to multiple back-end regions, there can be many AORs associated with the original transaction. Each of the multiple back-end regions can indicate they are switching control back to the front-end region (SWITCH LOCAL). Thus, with a 1-many mapping like this, there are many samples of the execution phase indicating switched requests—long enough to exceed 100% of the response time of other work completing in the service class.

Possible actions

None.

Example: fewer ended transactions with increased response times

The RMF workload activity report shows increased response times, and a decrease in the number of ended transactions.

Possible explanation

This situation could be caused by converting from ISC to MRO between the TOR and the AOR.

When two CICS regions are connected via VTAM intersystem communication (ISC) links, the perspective from a WLM viewpoint is that they behave differently from when they are connected via multiregion (MRO) option. One key difference is that, with ISC, both the TOR and the AOR are receiving a request from VTAM, so each believes it is starting and ending a given transaction. So for a given user request routed from the TOR via ISC to an AOR, there would be 2 completed transactions.

Let us assume they have response times of 1 second and .75 seconds respectively, giving for an average of .875 seconds. When the TOR routes via MRO, the TOR will describe a single completed transaction taking 1 second (in a begin-to-end phase), and the AOR will report it's .75 seconds as execution time. Therefore, converting from an ISC link to an MRO connection, for the same workload, could result in 1/2 the number of ended transactions and a corresponding increase in the response time reported by RMF.

Possible action

Increase CICS transaction goals prior to your conversion to an MRO connection.

Part 3. Analyzing the performance of a CICS system

This part gives an overview of performance analysis, identifies performance constraints, and describes various techniques for performance analysis.

Chapter 10. Overview of performance analysis

This chapter discusses performance analysis in the following sections:

There are four main uses for performance analysis:

1. You currently have no performance problems, but you simply want to adjust the system to give better performance, and you are not sure where to start.
2. You want to characterize and calibrate individual stand-alone transactions as part of the documentation of those transactions, and for comparison with some future time when, perhaps, they start behaving differently.
3. A system is departing from previously identified objectives, and you want to find out precisely where and why this is so. Although an online system may be operating efficiently when it is installed, the characteristics of the system usage may change and the system may not run so efficiently. This inefficiency can usually be corrected by adjusting various controls. At least some small adjustments usually have to be made to any new system as it goes live.
4. A system may or may not have performance objectives, but it appears to be suffering severe performance problems.

If you are in one of the first two categories, you can skip this chapter and the next and go straight to "Chapter 12. CICS performance analysis" on page 165.

If the current performance does ***not*** meet your needs, you should consider tuning the system. The basic rules of tuning are:

1. Identify the major constraints in the system.
2. Understand what changes could reduce the constraints, possibly at the expense of other resources. (Tuning is usually a trade-off of one resource for another.)
3. Decide which resources could be used more heavily.
4. Adjust the parameters to relieve the constrained resources.
5. Review the performance of the resulting system in the light of:
   - Your existing performance objectives
   - Progress so far
   - Tuning effort so far.
6. Stop if performance is acceptable; otherwise do one of the following:
   - Continue tuning
   - Add suitable hardware capacity
   - Lower your system performance objectives.

The tuning rules can be expressed in flowchart form as follows:

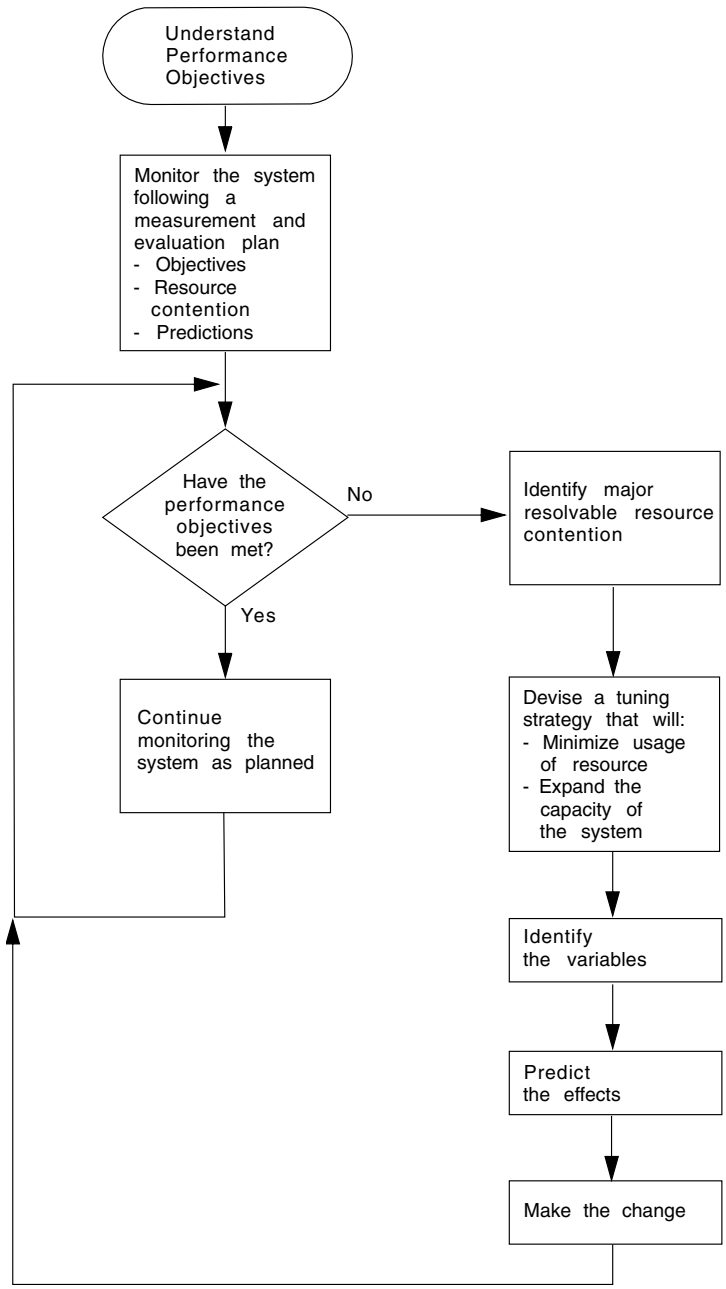


Figure 29. Flowchart to show rules for tuning performance

Establishing a measurement and evaluation plan

For some installations, a measurement and evaluation plan might be suitable. A measurement and evaluation plan is a structured way to measure, evaluate, and monitor the system's performance. By taking part in setting up this plan, the users, user management, and your own management will know how the system's performance is to be measured. In addition, you will be able to incorporate some of their ideas and tools, and they will be able to understand and concur with the plan, support you and feel part of the process, and provide you with feedback.

The implementation steps for this plan are:

1. Devise the plan

2. Review the plan
3. Implement the plan
4. Revise and upgrade the plan as necessary.

Major activities in using the plan are:

- Collect information periodically to determine:
  - Whether objectives have been met
  - Transaction activity
  - Resource utilization.
- Summarize and analyze the information. For this activity:
  - Plot volumes and averages on a chart at a specified frequency
  - Plot resource utilization on a chart at a specified frequency
  - Log unusual conditions on a daily log
  - Review the logs and charts weekly.
- Make or recommend changes if objectives have not been met.
- Relate past, current, and projected:
  - Transaction activity
  - Resource utilization.

  to determine:
  - If objectives continue to be met
  - When resources are being used beyond an efficient capacity.
- Keep interested parties informed by means of informal reports, written reports, and monthly meetings.

A typical measurement and evaluation plan might include the following items as objectives, with statements of recording frequency and the measurement tool to be used:

- Volume and response time for each department
- Network activity:
  - Total transactions
  - Tasks per second
  - Total by transaction type
  - Hourly transaction volume (total, and by transaction).
- Resource utilization examples:
  - DSA utilization
  - Processor utilization with CICS
  - Paging rate for CICS and for the system
  - Channel utilization
  - Device utilization
  - Data set utilization
  - Line utilization.
- Unusual conditions:
  - Network problems
  - Application problems
  - Operator problems
  - Transaction count for entry to transaction classes

- SOS occurrences
- Storage violations
- Device problems (not associated with the communications network)
- System outage
- CICS outage time.

Investigating the overall system

Always start by looking at the overall system before you decide that you have a specific CICS problem. The behavior of the system as a whole is usually just as important. You should check such things as total processor usage, DASD activity, and paging.

Performance degradation is often due to application growth that has not been matched by corresponding increases in hardware resources. If this is the case, solve the hardware resource problem first. You may still need to follow on with a plan for multiple regions.

Information from at least three levels is required:

1. *CICS*: Examine the CICS interval or end-of-day statistics for exceptions, queues, and other symptoms which suggest overloads on specific resources. A shorter reporting period can isolate a problem. Consider software as well as hardware resources: for example, utilization of VSAM strings or database threads as well as files and TP lines. Check run time messages sent to the console and to transient data destinations, such as CSMT and CSTL, for persistent application problems and network errors.

   Use tools such as CEMT and RMF, to monitor the online system and identify activity which correlates to periods of bad performance. Collect CICS monitoring facility history and analyze it, using tools like TIVOLI Performance Reporter to identify performance and resource usage exceptions and trends. For example, processor-intensive transactions which do little or no I/O should be noted. After they get control, they can monopolize the processor. This can cause erratic response in other transactions with more normally balanced activity profiles. They may be candidates for isolation in another CICS region.
2. *MVS*: Use SMF data to discover any relationships between periods of bad CICS performance and other concurrent activity in the MVS system. Use RMF data to identify overloaded devices and paths. Monitor CICS region paging rates to make sure that there is sufficient real storage to support the configuration.
3. *Network*: The proportion of response time spent in the system is usually small compared with transmission delays and queuing in the network. Use tools such as NetView, NPM, and VTAMPARS to identify problems and overloads in the network. Without automatic tools like these, you are dependent on the application users' subjective opinions that performance has deteriorated. This makes it more difficult to know how much worse performance has become and to identify the underlying reasons.

Within CICS, the performance problem is either a poor response time or an unexpected and unexplained high use of resources. In general, you need to look at the system in some detail to see why tasks are progressing slowly through the system, or why a given resource is being used heavily. The best way of looking at detailed CICS behavior is by using CICS auxiliary trace. But note that switching on auxiliary trace, though the best approach, may actually worsen existing poor performance while it is in use (see page 328).

The approach is to get a picture of task activity first, listing only the task traces, and then to focus on particular activities: specific tasks, or a very specific time interval. For example, for a response time problem, you might want to look at the detailed traces of one task that is observed to be slow. There may be a number of possible reasons.

The tasks may simply be trying to do too much work for the system. You are asking it to do too many things, it clearly takes time, and the users are simply trying to put too much through a system that can't do all the work that they want done.

Another possibility is that the system is real-storage constrained, and therefore the tasks progress more slowly than expected because of paging interrupts. These would show as delays between successive requests recorded in the CICS trace.

Yet another possibility is that many of the CICS tasks are waiting because there is contention for a particular function. There is a wait on strings on a particular data set, for example, or there is an application enqueue such that all the tasks issue an enqueue for a particular item, and most of them have to wait while one task actually does the work. Auxiliary trace enables you to distinguish most of these cases.

Other ways to analyze performance

Potentially, any performance measurement tool, including statistics and the CICS monitoring facility, may tell you something about your system that help in diagnosing problems. You should regard each performance tool as usable in some degree for each purpose: monitoring, single-transaction measurement, and problem determination.

Again, CICS statistics may reveal heavy use of some resource. For example, you may find a very large allocation of temporary storage in main storage, a very high number of storage control requests per task (perhaps 50 or 100), or high program use counts that may imply heavy use of program control LINK.

Both statistics and CICS monitoring may show exceptional conditions arising in the CICS run. Statistics can show waits on strings, waits for VSAM shared resources, waits for storage in GETMAIN requests, and so on. These also generate CICS monitoring facility exception class records.

While these conditions are also evident in CICS auxiliary trace, they may not appear so obviously, and the other information sources are useful in directing the investigation of the trace data.

In addition, you may gain useful data from the investigation of CICS outages. If there is a series of outages, common links between the outages should be investigated.

The next chapter tells you how to identify the various forms of CICS constraints, and Chapter 12 gives you more information on performance analysis techniques.

Chapter 11. Identifying CICS constraints

If current performance has been determined to be unacceptable, you need to identify the performance constraints (that is, the causes of the symptoms) so that they can be tuned. This chapter discusses these constraints in the following sections:

Major CICS constraints

Major constraints on a CICS system show themselves in the form of external symptoms: stress conditions and paging being the chief forms. This chapter describes these symptoms in some detail so that you can recognize them when your system has a performance problem, and know the ways in which CICS itself attempts to resolve various conditions.

The fundamental thing that has to be understood is that practically every symptom of poor performance arises in a system that is congested. For example, if there is a slowdown in DASD, transactions doing data set activity pile up: there are waits on strings; there are more transactions in the system, there is therefore a greater virtual storage demand; there is a greater real storage demand; there is paging; and, because there are more transactions in the system, the task dispatcher uses more processor power scanning the task chains. You then get task constraints, your MXT or transaction class limit is exceeded and adds to the processor overhead because of retries, and so on.

The result is that the system shows heavy use of ***all*** its resources, and this is the typical system stress. It does not mean that there is a problem with all of them; it means that there is a constraint that has yet to be found. To find the constraint, you have to find what is really affecting task life.

Response times

The basic criterion of performance in a production system is response time, but what is good response time? In straightforward data-entry systems, good response time implies subsecond response time. In normal production systems, good response time is measured in the five to ten second range. In scientific, compute-bound systems or in print systems, good response time can be one or two minutes.

Good performance, then, depends on a variety of factors including user requirements, available capacity, system reliability, and application design. Good performance for one system can be poor performance for another.

When checking whether the performance of a CICS system is in line with the system's expected or required capability, you should base this investigation on the hardware, software, and applications that are present in the installation.

If, for example, an application requires 100 accesses to a database, a response time of three to six seconds may be considered to be quite good. If an application requires only one access, however, a response time of three to six seconds for disk accesses would need to be investigated. Response times, however, depend on the speed of the processor, and on the nature of the application being run on the production system.

You should also observe how consistent the response times are. Sharp variations indicate erratic system behavior.

The typical way in which the response time in the system may vary with increasing transaction rate is gradual at first, then deteriorates rapidly and suddenly. The typical curve shows a sharp change when, suddenly, the response time increases dramatically for a relatively small increase in the transaction rate.

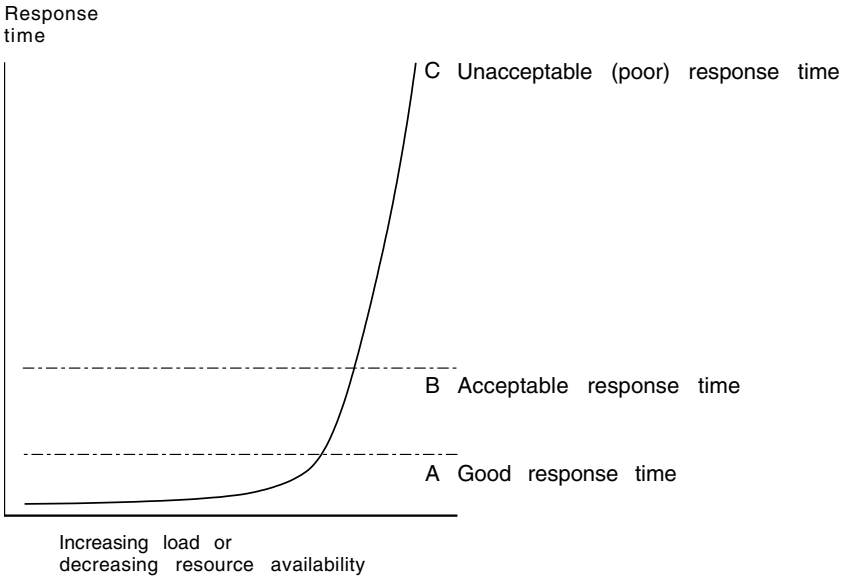


Figure 30. Graph to show the effect of response time against increasing load

For stable performance, it is necessary to keep the system operating below this point where the response time dramatically increases. In these circumstances, the user community is less likely to be seriously affected by the tuning activities being undertaken by the DP department, and these changes can be done in an unhurried and controlled manner.

Response time can be considered as being made up of queue time and service time. Service time is generally independent of usage, but queue time is not. For example, 50% usage implies a queue time approximately equal to service time, and 80% usage implies a queue time approximately four times the service time. If service time for a particular system is only a small component of the system

response, for example, in the processor, 80% usage may be acceptable. If it is a greater portion of the system response time, for example, in a communication line, 50% usage may be considered high.

If you are trying to find the response time from a terminal to a terminal, you should be aware that the most common "response time" obtainable from any aid or tool that runs in the host is the "internal response time." Trace can identify only when the software in the host, that is, CICS and its attendant software, first "sees" the message on the inbound side, and when it last "sees" the message on the outbound side.

Internal response time gives no indication of how long a message took to get from the terminal, through its control unit, across a line of whatever speed, through the communication controller (whatever it is), through the communication access method (whatever it is), and any delays before the channel program that initiated the read is finally posted to CICS. Nor does it account for the time it might take for CICS to start processing this input message. There may have been lots of work for CICS to do before terminal control regained control and before terminal control even found this posted event.

The same is true on the outbound side. CICS auxiliary trace knows when the application issued its request, but that has little to do with when terminal control found the request, when the access method ships it out, when the controllers can get to the device, and so on.

While the outward symptom of poor performance is overall bad response, there are progressive sets of early warning conditions which, if correctly interpreted, can ease the problem of locating the constraint and removing it.

In the advice given so far, we have assumed that CICS is the only major program running in your system. If batch programs or other online programs are running simultaneously with CICS, you must ensure that CICS receives its fair share of the system resources and that interference from other regions does not seriously degrade CICS performance.

Storage stress

Stress is the term used in CICS for a shortage of free space in one of the dynamic storage areas.

Storage stress can be a symptom of other resource constraints that cause CICS tasks to occupy storage for longer than is normally necessary, or of a flood of tasks which simply overwhelms available free storage, or of badly designed applications that require unreasonably large amounts of storage.

Controlling storage stress

Before CICS/ESA® Version 3, ***all*** non-resident, not-in-use programs were removed when a GETMAIN request could not be satisfied. Since CICS/ESA Version 3, storage stress has been handled as follows.

Nonresident, not-in-use programs may be deleted progressively with decreasing free storage availability as CICS determines appropriate, on a least-recently-used basis. The dispatching of new tasks is also progressively slowed as free storage approaches a critically small amount. This self-tuned activity tends to spread the

cost of managing storage. There may be more program loading overall, but the heavy overhead of a full program compression is not incurred at the critical time.

The loading or reloading of programs is handled by CICS with an MVS subtask. This allows other user tasks to proceed if a processor of the MVS image is available and even if a page-in is required as part of the program load.

User runtime control of storage usage is achieved through appropriate use of MXT and transaction class limits. This is necessary to avoid the short-on-storage condition that can result from unconstrained demand for storage.

Short-on-storage condition

CICS reserves a minimum number of free storage pages for use only when there is not enough free storage to satisfy an unconditional GETMAIN request even when all, not-in-use, nonresident programs have been deleted.

Whenever a request for storage results in the number of contiguous free pages in one of the dynamic storage areas falling below its respective cushion size, or failing to be satisfied even with the storage cushion, a cushion stress condition exists. Details are given in the storage manager statistics ("Times request suspended", "Times cushion released"). CICS attempts to alleviate the storage stress situation by releasing programs with no current user and slowing the attachment of new tasks. If these actions fail to alleviate the situation or if the stress condition is caused by a task that is suspended for SOS, a short-on-storage condition is signaled. This is accompanied by message DFHSM0131 or DFHSM0133.

Removing unwanted data set name blocks

One of the CICS dynamic storage areas, the ECDSA, is also used for data set name blocks, one of which is created for every data set opened by CICS file control. These DSN blocks are recovered at a warm or emergency restart. If you have an application that creates a large number of temporary data sets, all with a unique name, the number of DSN blocks can increase to such an extent that they can cause a short-on-storage condition.

If you have application programs that use temporary data sets, with a different name for every data set created, it is important that your programs remove these after use. See the *CICS System Programming Reference* for information about how you can use the SET DSNAME command to remove unwanted temporary data sets from your CICS regions.

LE run time options for AMODE(24) programs

The default LE run time options for CICS are (among other things) ALL31(ON) and STACK(ANY). This means that all programs must run above the line (AMODE(31)) in an LE environment. To allow AMODE(24) programs to run in an LE environment, ALL31(OFF) and STACK(BELOW) can be specified. However, if you globally change these options so that all programs can use them, a lot of storage will be put below the line, which can cause a short-on-storage condition.

Purging of tasks

If a CICS task is suspended for longer than its DTIMOUT value, it may be purged if SPURGE=YES is specified on the RDO transaction definition. That is, the task is abended and its resources freed, thus allowing other tasks to use those resources. In this way, CICS attempts to resolve what is effectively a deadlock on storage.

CICS hang

If purging tasks is not possible or not sufficient to solve the problem, CICS ceases processing. You must then either cancel and restart the CICS system, or initiate or allow an XRF takeover.

Effect of program loading on CICS

CICS employs MVS load under an MVS subtask to load programs. This provides the benefits, relative to versions of CICS prior to CICS Transaction for OS/390 Release 1, of fast loading from DASD and allows the use of the library lookaside function of MVS to eliminate most DASD I/Os by keeping copies of programs in an MVS controlled dataspace exploiting expanded storage.

A page-in operation causes the MVS task which requires it to stop until the page has been retrieved. If the page is to be retrieved from DASD, this has a significant effect. When the page can be retrieved from expanded storage, the impact is only a relatively small increase in processor usage.

The loading of a program into CICS storage can be a major cause of page-ins. Because this is carried out under a subtask separate from CICS main activity, such page-ins do not halt most other CICS activities.

What is paging?

The virtual storage of a processor may far exceed the size of the central storage available in the configuration. Any excess must be maintained in auxiliary storage (DASD), or in expanded storage. This virtual storage occurs in blocks of addresses called “pages”. Only the most recently referenced pages of virtual storage are assigned to occupy blocks of physical central storage. When reference is made to a page of virtual storage that does not appear in central storage, the page is brought in from DASD or expanded storage to replace a page in central storage that is not in use and least recently used.

The newly referenced page is said to have been “paged in”. The displaced page may need to be “paged out” if it has been changed.

Paging problems

It is the *page-in* rate that is of primary concern, because page-in activity occurs synchronously (that is, an MVS task stops until the page fault is resolved). Page-out activity is overlapped with CICS processing, so it does not appreciably affect CICS throughput.

A page-in from expanded storage incurs only a small processor usage cost, but a page-in from DASD incurs a time cost for the physical I/O and a more significant increase in processor usage.

Thus, extra DASD page-in activity slows down the rate at which transactions flow through the CICS system, that is, transactions take longer to get through CICS, you get more overlap of transactions in CICS, and so you need more virtual and real storage.

If you suspect that a performance problem is related to excessive paging, you can use RMF to obtain the paging rates.

Consider controlling CICS throughput by using MXT and transaction class limits in CICS on the basis that a smaller number of concurrent transactions requires less real storage, causes less paging, and may be processed faster than a larger number of transactions.

When a CICS system is running with transaction isolation active, storage is allocated to user transactions in multiples of 1MB. This means that the virtual storage requirement for a CICS system with transaction isolation enabled is very large. This does not directly affect paging which only affects those 4K byte pages that have been touched. More real storage is required in ELSQA, however, and for more information on transaction isolation and real storage see "Transaction isolation and real storage requirements" on page 298.

What is an ideal CICS paging rate from DASD? Less than one page-in per second is best to maximize the throughput capacity of the CICS region. Anything less than five page-ins per second is probably acceptable; up to ten may be tolerable. Ten per second is marginal, more is probably a major problem. Because CICS performance can be affected by the waits associated with paging, you should not allow paging to exceed more than five to ten pages per second.

Note: The degree of sensitivity of CICS systems to paging from DASD depends on the transaction rate, the processor loading, and the average internal lifetime of the CICS tasks. An ongoing, hour-on-hour rate of even five page-faults per second may be excessive for some systems, particularly when you realize that peak paging rates over periods of ten seconds or so could easily be four times that figure.

What paging rates are excessive on various processors and are these rates operating-system dependent? Excessive paging rates should be defined as those which cause excessive delays to applications. The contribution caused by the high-priority paging supervisor executing instructions and causing applications to wait for the processor is probably a minor consideration as far as overall delays to applications are concerned. Waiting on a DASD device is the dominant part of the overall delays. This means that the penalty of "high" paging rates has almost nothing to do with the processor type.

CICS systems are usually able to deliver much better response times with somewhat better processor utilization when the potential of large amounts of central and expanded storage is exploited by keeping more data and programs in memory.

Recovery from storage violation

CICS can detect storage violations when:

- The duplicate storage accounting area (SAA) or the initial SAA of a TIOA storage element has become corrupted.
- The leading storage check zone or the trailing storage check zone of a user task storage has become corrupted.

A storage violation can occur in two basic situations:

1. When CICS detects an error during its normal processing of a FREEMAIN request for an individual element of a TIOA storage, and finds that the two storage check zones of the duplicate SAA and the initial SAA are not identical.

2. CICS also detects user violations involving user task storage by checking the storage check zones of an element of user task storage following a FREEMAIN command.

When a storage violation is detected, an exception trace entry is made in the internal trace table. A message (DFHSM0102) is issued and a CICS system dump follows if the dump option is switched on.

Storage violations can be reduced considerably if CICS has storage protection, and transaction isolation, enabled.

See the *CICS Problem Determination Guide* for further information about diagnosing and dealing with storage violations.

Dealing with limit conditions

The main limit conditions or constraints that can occur in a CICS system include those listed at the beginning of this chapter. Stress conditions generally tell you that certain limiting conditions have been reached. If these conditions occur, additional processing is required, and the transactions involved have to wait until resources are released.

To summarize, limit conditions can be indicated by the following:

- Virtual storage conditions ("short-on-storage": SOS). This item in the CICS storage manager statistics shows a deficiency in the allocation of virtual storage space to the CICS region.

  In most circumstances, allocation of more virtual storage does not in itself cause a degradation of performance. You should determine the reason for the condition in case it is caused by some form of error. This could include failure of applications to free storage (including temporary storage), unwanted multiple copies of programs or maps, storage violations, and high activity of nonresident exception routines caused by program or hardware errors.

  All new applications should be written to run above the 16MB line. The dynamic storage areas above the 16MB line can be expanded up to the 2GB limit of 31-bit addressing. The dynamic storage areas below the 16MB line are limited to less than the region size, which is less than 16MB.
- Number of simultaneous tasks (MXT and transaction class limit) reached (shown in the transaction manager statistics).
- Maximum number of VTAM receive-any RPLs in use (shown in the VTAM statistics).
- 'Wait-on-string' and associated conditions for VSAM data sets (shown in the file control statistics).

Check how frequently the limit conditions occur. In general:

- If *no* limit conditions occur, this implies that too many resources have been allocated. This is quite acceptable if the resource is inexpensive, but not if the resource is both overallocated and of more use elsewhere.
- *Infrequent* occurrence of a limit condition is an indication of good usage of the particular resource. This usually implies a healthy system.
- *Frequent* occurrence (greater than 5% of transactions) usually reveals a problem, either directly or indirectly, that needs action to prevent more obvious signs of poor performance. If the frequency is greater than about 10%, you may have to

take some action quickly because the actions taken by CICS itself (dynamic program storage compression, release of storage cushion, and so on) can have a perceptible effect on performance.

Your own actions should include:

- Checking for errors
- Raising the limit, provided that it does not have a degrading effect on other areas
- Allocating more resources to remove contention
- Checking recovery usage for contention.

Identifying performance constraints

When you are dealing with limit conditions, you may find it helpful to check the various points where performance constraints can exist in a system. These points are summarized below under hardware and software constraints.

Hardware constraints

1. *Processor cycles*. It is not uncommon for transactions to execute more than one million instructions. To execute these instructions, they must contend with other tasks and jobs in the system. At different times, these tasks must wait for such activities as file I/O. Transactions give up their use of the processor at these points and must contend for use of the processor again when the activity has completed. Dispatching priorities affect which transactions or jobs get use of the processor, and batch or other online systems may affect response time through receiving preferential access to the processor. Batch programs accessing online databases also tie up those databases for longer periods of time if their dispatching priority is low. At higher usages, the wait time for access to the processor can be significant.
2. *Real storage (working set)*. Just as transactions must contend for the processor, they also must be given a certain amount of real storage. A real storage shortage can be particularly significant in CICS performance because a normal page fault to acquire real storage results in synchronous I/O. The basic design of CICS is asynchronous, which means that CICS processes requests from multiple tasks concurrently to make maximum use of the processor. Most paging I/O is synchronous and causes the MVS task that CICS is using to wait, and that part of CICS cannot do any further processing until the page operation completes. Most, but not all, of CICS processing uses a single MVS task (called 'QUASI' in the dispatcher statistics).
3. *Database-associated hardware (I/O) contention*. When data is being accessed to provide information that is required in a transaction, an I/O operation passes through the processor, the processor channel, a disk control unit, the head of string on a string of disks, and the actual disk device where the data resides. If any of these devices are overused, the time taken to access the data can increase significantly. This overuse can be the result of activity on one data set, or on a combination of active data sets. Error rates also affect the usage and performance of the device. In shared DASD environments, contention between processors also affects performance. This, in turn, increases the time that the transaction ties up real and virtual storage and other resources.

   The use of large amounts of central and expanded storage by using very large data buffers, and by keeping programs in storage, can significantly reduce DB I/O contention and somewhat reduce processor utilization while delivering significant internal response time benefits.

4. *Network-associated hardware contention*. The input and output messages of a transaction must pass from the terminal to a control unit, a communications link, a network controller, a processor channel, and finally the processor. Just as overuse of devices to access data can affect response time, so excessive use of network resources can cause performance degradation. Error rates affect performance as well. In some cases, the delivery of the output message is a prerequisite to freeing the processor resources that are accessed, and contention can cause these resources to be tied up for longer periods.

Software constraints

1. *Database design*. A data set or database needs to be designed to the needs of the application it is supporting. Such factors as the pattern of access to the data set (especially whether it is random or sequential), access methods chosen, and the frequency of access determine the best database design. Such data set characteristics as physical record size, blocking factors, the use of alternate or secondary indexes, the hierarchical or relational structure of database segments, database organization (HDAM, HIDAM, and so on), and pointer arrangements are all factors in database performance.

   The length of time between data set reorganizations can also affect performance. The efficiency of accesses decreases as the data set becomes more and more fragmented. This fragmentation can be kept to the minimum by reducing the length of time between data set reorganizations.
2. *Network design*. This item can often be a major factor in response time because the network links are much slower than most components of an online system. Processor operations are measured in nanoseconds, line speeds in seconds. Screen design can also have a significant effect on overall response time. A 1200-byte message takes one second to be transmitted on a relatively high-speed 9600 bits-per-second link. If 600 bytes of the message are not needed, half a second of response time is wasted. Besides screen design and size, such factors as how many terminals are on a line, the protocols used (SNA, bisynchronous), and full-or half-duplex capabilities can affect performance.
3. *Use of specific software interfaces or serial functions*. The operating system, terminal access method, database manager, data set access method, and CICS must all communicate in the processing of a transaction. Only a given level of concurrent processing can occur at these points, and this can also cause a performance constraint. Examples of this include the VTAM receive any pool (RAPOOL), VSAM data set access (strings), CICS temporary storage, CICS transient data, and CICS intercommunication sessions. Each of these can have a single or multiserver queueing effect on a transaction's response time, and can tie up other resources by slowing task throughput.

One useful technique for isolating a performance constraint in a CICS system with VTAM is to use the IBMTEST command issued from a user's terminal. This terminal must not be in session with CICS, but must be connected to VTAM.

You enter at a VTAM terminal:

```
IBMTEST (n)(,data)
```

where *n* is the number of times you want the data echoed, and *data* may consist of any character string. If you enter no data, the alphabet and the numbers zero through nine are returned to the terminal. This command is responded to by VTAM.

IBMTEST is an echo test designed to give the user a rough idea of the VTAM component of terminal response time. If the response time is fast in a slow-response system, the constraint is not likely to be any component from VTAM onward. If this response is slow, VTAM or the network may be the reason. This sort of deductive process in general can be useful in isolating constraints.

To avoid going into session with CICS, you may have to remove APPLID= from the LU statement or CONNECT=AUTO from the TERMINAL definition.

Resource contention

The major resources used or managed by CICS consist of the following:

- Processor
- Real storage
- Virtual storage
- Software (specification limits)
- Channels
- Control units
- Lines
- Devices
- Sessions to connected CICS systems.

Contention at lower levels prevents full use of higher-level resources. To avoid or reduce resource contention, you can:

- Minimize or eliminate the use of a resource by:
  - Reordering, relocating, or reducing its size
  - Redesign, rewriting, rescheduling, or reducing processing time
  - Education, eliminating a function, or controlling its usage.
- Give the resource more capacity
- Exchange one resource with another:
  - Processor with virtual storage
  - Real storage with paging I/O
  - Paging I/O with program library I/O
  - Priorities of various end-users with each other
  - CICS response times with batch throughput
  - Batch throughput with more DP operators.

Two sets of symptoms and solutions are provided in this chapter. The first set provides suggested solutions for poor response, and the second set provides suggested solutions for a variety of resource contention problems.

Solutions for poor response time

Table 10 shows four levels of response time, in decreasing order of severity. The major causes are shown for each level together with a range of suggested solutions. Your first step is to check the causes by following the advice given in “Chapter 12. CICS performance analysis” on page 165. When you have identified the precise causes, the relevant checklist in “Chapter 14. Performance checklists” on page 177 tells you what solutions are available and where to find information in Part 4 of this book on how to implement the solutions.

Table 10. CICS response time checklist

Level	Symptom	Major Causes	Overall Solution
1	Poor response at all loads for all transactions	High level of paging	Reduce working set, or allocate more real storage
		Very high usage of major resources	Reconsider system resource requirements and redesign system Check for application errors and resource contention
2	Poor response at medium and high loads	High level of paging	Reduce working set, or allocate more real storage
		High processor usage	Reduce pathlength, or increase processor power
		High DB or data set usage	Reorganize data sets, or reduce data transfer, or increase capacity
		High communication network usage	Reduce data transfer, or increase capacity
		TP or I/O access-method constraint	Increase buffer availability
		CICS limit values exceeded	Change operands, or provide more resources, or check if errors in application
3	Poor response for certain transactions only	Identify common characteristics	As for level 2
		Lines or terminal usage	Increase capacity, or reduce data transfer, or change transaction logic
		Data set usage	Change data set placement buffer allocations or change enqueue logic or data set design
		High storage usage	Redesign or tune applications
		Same subprograms used by transactions	Redesign or tune application subprograms
		Same access method or CICS features used by transactions	Reallocate resource or change application. Reevaluate use of feature in question
		Limit conditions	Reallocate resource or change application
4	Poor response for certain terminals	Check network loading as appropriate	Increase capacity of that part of network
		Check operator techniques	Revise terminal procedures
		Check CEDA terminal definitions	Redefine CEDA terminal definitions

Symptoms and solutions for resource contention problems

This section presents a general range of solutions for each type of constraint. You should:

1. Confirm that your diagnosis of the type of constraint is correct, by means of detailed performance analysis. "Chapter 12. CICS performance analysis" on page 165 describes various techniques.
2. Read "Chapter 13. Tuning the system" on page 173 for general advice on performance tuning.
3. See the relevant sections in Part 4 of this book for detailed information on applying the various solutions.
4. Improve virtual storage exploitation. This requires:
   - Large data buffers above the 16MB line or in Hiperspace
   - Programs that run above the 16MB line
   - Large amounts of central and expanded storage to support the virtual storage exploitation.

   Such a system can deliver better internal response times, while minimizing DASD I/O constraint and reducing processor utilization.

DASD constraint

Symptoms

- Slow response times (the length of the response time depends on the number of I/O operations, with a longer response time when batch mode is active)
- High DSA utilization
- High paging rates
- MXT limit frequently reached
- SOS condition often occurs.

Solutions

- Reduce the number of I/O operations
- Tune the remaining I/O operations
- Balance the I/O operations load.

  See "DASD tuning" on page 194 for suggested solutions.

Communications network constraint

Symptoms

- Slow response times
- Good response when few terminals are active on a line, but poor response when many terminals are active on that line
- Big difference between internal response time and terminal response time.

Solutions

- Reduce the line utilization.
- Reduce delays in data transmission.
- Alter the network.

Remote systems constraints

Symptoms

- SOS condition or MXT occur when there is a problem with a connected region.
- CICS takes time to recover when the problem is fixed.

Solutions

- Control the amount of queuing which takes place for the use of the connections to the remote systems.
- Improve the response time of the remote system.

Virtual storage constraint

Symptoms

- Slow response times
- Multiple loads of the same program
- Increased I/O operations against program libraries
- High paging rates
- SOS condition often occurs.

Solutions

- Tune the MVS system to obtain more virtual storage for CICS (increase the region size).
- Expand or make more efficient use of the dynamic storage area.

See the "Virtual storage above and below 16MB line checklist" on page 178 for a detailed list of suggested solutions.

Real storage constraint

Symptoms

- High paging rates
- Slow response times
- MXT limit frequently reached
- SOS condition often occurs.

Solutions

- Reduce the demands on real storage
- Tune the MVS system to obtain more real storage for CICS
- Obtain more central and expanded storage.

See the "Real storage checklist" on page 179 for a detailed list of suggested solutions.

Processor cycles constraint

Symptoms

- Slow response times
- Low-priority transactions respond very slowly
- Low-priority work gets done very slowly.

Solutions

- Increase the dispatching priority of CICS.
- Reevaluate the relative priorities of operating system jobs.
- Reduce the number of MVS regions (batch).
- Reduce the processor utilization for productive work.
- Use only the CICS facilities that you really require.
- Turn off any trace that is not being used.
- Minimize the data being traced by reducing the:
  - Scope of the trace
  - Frequency of running trace.
- Obtain a faster processor.

See the "Processor cycles checklist" on page 180 for a detailed list of suggested solutions.

Chapter 12. CICS performance analysis

This chapter describes aspects of CICS performance analysis in the following:

Performance analysis, as compared with monitoring, is the use of certain performance tools described in Part 2 to:

- Investigate a deviation from performance objectives that is resulting in performance deterioration, and identify performance problems
- Identify where a system can be adjusted to give a required level of performance
- Characterize and calibrate individual stand-alone transactions as part of the documentation of those transactions, and for comparison with some future time when, perhaps, they start behaving differently.

Assessing the performance of a DB/DC system

You may find the following performance measurements helpful in determining the performance of a system:

1. *Processor usage*: This item reflects how active the processor is. Although the central processor is of primary concern, 37X5 communications controllers and terminal control units (these can include an intelligent cluster controller such as the 3601 and also the 3270 cluster control units) can also increase response time if they are heavily used.
2. *I/O rates*: These rates measure the amount of access to a disk device or data set over a given period of time. Again, acceptable rates vary depending on the speed of the hardware and response time requirements.
3. *Terminal message or data set record block sizes*: These factors, when combined with I/O rates, provide information on the current load on the network or DASD subsystem.
4. *Indications of internal virtual storage limits*: These vary by software component, including storage or buffer expansion counts, system messages, and program abends because of system stalls. In CICS, program fetches on nonresident programs and system short-on-storage or stress messages reflect this condition.
5. *Paging rates*: CICS can be sensitive to a real storage shortage, and paging rates reflect this shortage. Acceptable paging to DASD rates vary with the speed of the DASD and response time criteria. Paging rates to expanded storage are only as important as its effect on processor usage.
6. *Error rates*: Errors can occur at any point in an online system. If the errors are recoverable, they can go unnoticed, but they put an additional load on the resource on which they are occurring.

You should investigate both system conditions and application conditions.

System conditions

A knowledge of these conditions enables you evaluate the performance of the system as a whole:

- System transaction rate (average and peak)
- Internal response time and terminal response time, preferably compared with transaction rate
- Working set, at average and peak transaction rates
- Average number of disk accesses per unit time (total, per channel, and per device)
- Processor usage, compared with transaction rate
- Number of page faults per second, compared with transaction rate and real storage
- Communication line usage (net and actual)
- Average number of active CICS tasks
- Number and duration of outages.

Application conditions

These conditions, measured both for individual transaction types and for the total system, give you an estimate of the behavior of individual application programs.

You should gather data for each main transaction and average values for the total system. This data includes:

- Program calls per transaction
- CICS storage GETMAINs and FREEMAINs (number and amount)
- Application program and transaction usage
- File control (data set, type of request)
- Terminal control (terminal, number of inputs and outputs)
- Transaction routing (source, target)
- Function shipping (source, target)
- Other CICS requests.

Methods of performance analysis

You can use two methods for performance analysis:

1. Measuring a system under full production load (*full-load* measurement), to get all information that is measurable only under high system-loading.
2. Measuring single-application transactions (*single-transaction* measurement), during which the system should not carry out any other activities. This gives an insight into the behavior of single transactions under optimum system conditions.

Because a system can have a variety of problems, we cannot recommend which option you should use to investigate the behavior of a system. When in doubt about the extent of a problem, you should always use both methods.

Rapid performance degradation often occurs after a threshold is exceeded and the system approaches its ultimate load. You can see various indications only when the system is fully loaded (for example, paging, short-on-storage condition in CICS, and so on), and you should usually plan for a full-load measurement.

Bear in mind that the performance constraints might possibly vary at different times of the day. You might want to run a particular option that puts a particular pressure on the system only at a certain time in the afternoon.

If a full-load measurement reveals no serious problems, or if a system is not reaching its expected performance capability under normal operating conditions, you can then use single-transaction measurement to reveal how individual system transactions behave and to identify the areas for possible improvement.

Often, because you have no reliable information at the beginning of an investigation into the probable causes of performance problems, you have to examine and analyze the whole system.

Before carrying out this analysis, you must have a clear picture of the functions and the interactions of the following components:

- Operating system supervisor with the appropriate access methods
- CICS management modules and control tables
- VSAM data sets
- DL/I databases
- DB2
- External security managers
- Performance monitors
- CICS application programs
- Influence of other regions
- Hardware peripherals (disks and tapes).

In addition, you should collect the following information:

- Does performance fluctuate or is it uniformly bad?
- Are performance problems related to a specific hour, day, week, or month?
- Has anything in the system been changed recently?
- Have all such changes been fully documented?

Full-load measurement

A full-load measurement highlights latent problems in the system. It is important that full-load measurement lives up to its name, that is, you should make the measurement when, from production experience, the peak load is reached. Many installations have a peak load for about one hour in the morning and again in the afternoon. CICS statistics and various performance tools can provide valuable information for full-load measurement. In addition to the overall results of these tools, it may be useful to have the CICS auxiliary trace or RMF active for about one minute.

CICS auxiliary trace

CICS auxiliary trace can be used to find situations that occur under full load. For example, all ENQUEUEs that cannot immediately be honored in application programs result in a suspension of the issuing task. If this frequently happens, attempts to control the system by using the CEMT master transaction, are not effective.

Trace is a very heavy overhead. Use trace selectivity options to minimize this overhead.

RMF

It is advisable to do the RMF measurement without any batch activity. (See "Resource measurement facility (RMF)" on page 25 for a detailed description of this tool. Guidance on how to use RMF with the CICS monitoring facility is given in "Using CICS monitoring SYSEVENT information with RMF" on page 63.)

For full-load measurement, the system activity report and the DASD activity report are important.

The most important values for full-load measurement are:

- Processor usage
- Channel and disk usage
- Disk unit usage
- Overlapping of processor with channel and disk activity
- Paging
- Count of start I/O operations and average start I/O time
- Response times
- Transaction rates.

You should expect stagnant throughput and sharply climbing response times as the processor load approaches 100%.

It is difficult to forecast the system paging rate that can be achieved without serious detriment to performance, because too many factors interact. You should observe the reported paging rates; note that short-duration severe paging leads to a rapid increase in response times.

In addition to taking note of the count of start I/O operations and their average length, you should also find out whether the system is waiting on one device only. With disks, for example, it can happen that several frequently accessed data sets are on one disk and the accesses interfere with each other. In each case, you should investigate whether a system wait on a particular unit could not be minimized by reorganizing the data sets.

The RMF DASD activity report includes the following information:

- A summary of all disk information
- Per disk, a breakdown by system number and region
- Per disk, the distribution of the seek arm movements
- Per disk, the distribution of accesses with and without arm movement.

Use IOQ(DASD) option in RMF monitor 1 to show DASD control unit contention.

After checking the relationship of accesses with and without arm movement, for example, you may want to move to separate disks those data sets that are periodically very frequently accessed.

Comparison charts

You might wish to consider using a comparison chart to measure key aspects of your system's performance before and after tuning changes have been made. A suggested chart is as follows:

Table 11. Comparison chart

Run:					
DL/I transactions	Number				
	Response				
VSAM transactions	Number				
	Response				
Response times	DL/I				
	VSAM				
Most heavily used transaction	Number				
	Response				
Average-use transaction	Number				
	Response				
Paging rate	System				
	CICS				
DSA virtual storage	Maximum				
	Average				
Tasks	Peak				
	At MXT				
Most heavily used DASD	Response				
	Utilization				
Average-use DASD	Response				
	Utilization				
CPU utilization					

The use of this type of comparison chart requires the use of TPNS, RMF, and CICS interval statistics running together for about 20 minutes, at a peak time for your system. It also requires you to identify the following:

- A representative selection of terminal-oriented DL/I transactions accessing DL/I databases
- A representative selection of terminal-oriented transactions processing VSAM files
- The most heavily used transaction
- Two average-use nonterminal-oriented transactions writing data to intrapartition transient data destinations
- The most heavily used volume in your system
- A representative average-use volume in your system.

To complete the comparison chart for each CICS run before and after a tuning change, you can obtain the figures from the following sources:

- *DL/I transactions*: you should first identify a selection of terminal-oriented DL/I transactions accessing DL/I databases.
- *VSAM transactions*: similarly, you should first identify a selection of terminal-oriented transactions processing VSAM files.

- *Response times*: external response times are available from the TPNS terminal response time analysis report; internal response times are available from RMF. The "DL/I" subheading is the average response time calculated at the 99th percentile for the terminal-oriented DL/I transactions you have previously selected. The "VSAM" subheading is the average response time calculated at the 99th percentile for the terminal-oriented VSAM transactions you have previously selected.
- *Paging rate (system)*: this is from the RMF paging activity report, and is the figure shown for total system non-VIO non-swap page-ins added to the figure shown for the total system non-VIO non-swap page-outs. This is the total paging rate per second for the entire system.
- *Tasks*: this is from the transaction manager statistics (part of the CICS interval, end-of-day, and requested statistics). The "Peak" subheading is the figure shown for "Peak Number of Tasks" in the statistics. The "At MXT" subheading is the figure shown for "Number of Times at Max. Task" in the statistics.
- *Most heavily used DASD*: this is from the RMF direct access device activity report, and relates to the most heavily used volume in your system. The "Response" subheading is the figure shown in the "Avg. Resp. Time" column for the volume you have selected. The "Utilization" subheading is the figure shown in the "% Dev. Util." column for that volume.
- *Average-use DASD*: this is also from the RMF direct access device activity report, and relates to a representative average-use volume in your system. The "Response" subheading is the figure shown in the "Avg. Resp. Time" column for the volume you have selected. The "Utilization" subheading is the figure shown in the "% Dev. Util." column for that volume.
- *Processor utilization*: this is from the RMF processor activity report.

This chart is most useful when comparing before-and-after changes in performance while you are tuning your CICS system.

Single-transaction measurement

You can use full-load measurement to evaluate the average loading of the system per transaction. However, this type of measurement cannot provide you with information on the behavior of a single transaction and its possible excessive loading of the system. If, for example, nine different transaction types issue five start I/Os (SIOs) each, but the tenth issues 55 SIOs, this results in an average of ten SIOs per transaction type. This should not cause concern if they are executed simultaneously. However, an increase of the transaction rate of the tenth transaction type could possibly lead to poor performance overall.

Sometimes, response times are quite good with existing terminals, but adding a few more terminals leads to unacceptable degradation of performance. In this case, the performance problem may be present with the existing terminals, and has simply been highlighted by the additional load.

To investigate this type of problem, do a full-load measurement as well as a single-transaction measurement. To be of any use, the single-transaction measurement must be done when no batch region is running, and there must be no activity in CICS apart from the test screen. Even the polling of remote terminals should be halted.

You should measure each existing transaction that is used in a production system or in a final test system. Test each transaction two or three times with different

data values, to exclude an especially unfavorable combination of data. Document the sequence of transactions and the values entered for each test as a prerequisite for subsequent analysis or interpretation.

Between the tests of each single transaction, there should be a pause of several seconds, to make the trace easier to read. A copy of the production database or data set should be used for the test, because a test data set containing 100 records can very often result in completely different behavior when compared with a production data set containing 100 000 records.

The condition of data sets has often been the main reason for performance degradation, especially when many segments or records have been added to a database or data set. Do not do the measurements directly after a reorganization, because the database or data set is only in this condition for a short time. On the other hand, if the measurement reveals an unusually large number of disk accesses, you should reorganize the data and do a further measurement to evaluate the effect of the data reorganization.

You may feel that single-transaction measurement under these conditions with only one terminal is not an efficient tool for revealing a performance degradation that might occur when, perhaps 40 or 50 terminals are in use. Practical experience has shown, however, that this is usually the only means for revealing and rectifying, with justifiable expense, performance degradation under full load. The main reason for this is that it is sometimes a single transaction that throws the system behavior out of balance. Single-transaction measurement can be used to detect this.

Ideally, single-transaction measurement should be carried out during the final test phase of the transactions. This gives the following advantages:

- Any errors in the behavior of transactions may be revealed before production starts, and these can be put right during validation, without loading the production system unnecessarily.
- The application is documented during the measurement phase. This helps to identify the effects of later changes.

CICS auxiliary trace

Auxiliary trace is a standard feature of CICS, and gives an overview of transaction flows so that you can quickly and effectively analyze them.

From this trace, you can find out whether a specified application is running as it is expected to run. In many cases, it may be necessary for the application programmer responsible to be called in for the analysis, to explain what the transaction should actually be doing.

If you have a very large number of transactions to analyze, you can select, in a first pass, the transactions whose behavior does not comply with what is expected.

If all transactions last much longer than expected, this almost always indicates a system-wide error in application programming or in system implementation. The analysis of a few transactions is then sufficient to determine the error.

If, on the other hand, only a few transactions remain in this category, these transactions should be analyzed next, because it is highly probable that most performance problems to date arise from these.

Chapter 13. Tuning the system

When you have identified specific constraints, you will have identified the system resources that need to be tuned. The three major steps in tuning a system are:

1. Determine acceptable tuning trade-offs
2. Make the change to the system
3. Review the results of tuning.

Determining acceptable tuning trade-offs

The art of tuning can be summarized as finding and removing constraints. In most systems, the performance is limited by a single constraint. However, removing that constraint, while improving performance, inevitably reveals a different constraint, and you might often have to remove a series of constraints. Because tuning generally involves decreasing the load on one resource at the expense of increasing the load on a different resource, relieving one constraint always creates another.

A system is always constrained. You do not simply remove a constraint; you can only choose the most satisfactory constraint. Consider which resources can accept an additional load in the system without themselves becoming worse constraints.

Tuning usually involves a variety of actions that can be taken, each with its own trade-off. For example, if you have determined virtual storage to be a constraint, your tuning options may include reducing buffer allocations for data sets, or reducing terminal scan delay (ICVTSD) to shorten the task life in the processor.

The first option increases data set I/O activity, and the second option increases processor usage. If one or more of these resources are also constrained, tuning could actually cause a performance degradation by causing the other resource to be a greater constraint than the present constraint on virtual storage.

Making the change to the system

The next step in the tuning process is to make the actual system modifications that are intended to improve performance. You should consider several points when adjusting the system:

- Tuning is the technique of making small changes to the system's resource allocation and availability to achieve relatively large improvements in response time.
- Tuning is not always effective. If the system response is too long and all the system resources are lightly used, you see very little change in the CICS response times. (This is also true if the wrong resources are tuned.) In addition, if the constraint resource, for example, line capacity, is being fully used, the only solution is to provide more capacity or redesign the application (to transmit less data, in the case of line capacity).
- Do not tune just for the sake of tuning. Tune to relieve identified constraints. If you tune resources that are not the primary cause of performance problems, this has little or no effect on response time until you have relieved the major constraints, and it may actually make subsequent tuning work more difficult. If there is any significant improvement potential, it lies in improving the performance of the resources that ***are*** major factors in the response time.

- In general, tune major constraints first, particularly those that have a significant effect on response time. Arrange the tuning actions so that items having the greatest effect are done first. In many cases, one tuning change can solve the performance problem if it addresses the cause of the degradation. Other actions may then be unnecessary. Further, improving performance in a major way can alleviate many user complaints and allow you to work in a more thorough way. The 80/20 rule applies here; a small number of system changes normally improves response time by most of the amount by which it can be improved, assuming that those changes address the main causes of performance problems.
- Make one tuning change at a time. If two changes are made at the same time, their effects may work in opposite directions and it may be difficult to tell which of them had a significant effect.
- Change allocations or definitions gradually. For example, when reducing the number of resident programs in a system, do not change all programs in a system from RES=YES to RES=NO at once. This could cause an unexpected lengthening of response times by increasing storage usage because of fragmentation, and increasing processor usage because of higher program loading activity. If you change a few programs at a time, starting with the lesser-used programs, this can give you a better idea of the overall results.

  The same rule holds true for buffer and string settings and other data set operands, transaction and program operands, and all resources where the operand can be specified individually for each resource. For the same reason, do not make large increases or decreases in the values assigned to task limits such as MXT.
- Continue to monitor constraints during the tuning process. Because each adjustment changes the constraints in a system, these constraints vary over time. If the constraint changes, tuning must be done on the new constraint because the old one is no longer the restricting influence on performance. In addition, constraints may vary at different times during the day.
- Put fallback procedures in place before starting the tuning process. As noted earlier, some tuning can cause unexpected performance results. If this leads to poorer performance, it should be reversed and something else tried. If previous definitions or path designs were not saved, they have to be redefined to put the system back the way it was, and the system continues to perform at a poorer level until these restorations are made. If the former setup is saved in such a way that it can be recalled, back out of the incorrect change becomes much simpler.

Reviewing the results of tuning

After each adjustment has been done, review the performance measurements that have been identified as the performance problem to verify that the desired performance changes have occurred and to quantify that change. If performance has improved to the point that service level agreements are being met, no more tuning is required. If performance is better, but not yet acceptable, investigation is required to determine the next action to be taken, and to verify that the resource that was tuned is still a constraint. If it is not still a constraint, new constraints need to be identified and tuned. This is a return to the first step of the tuning process, and you should repeat the next steps in that process until an acceptable performance level is reached.

Part 4. Improving the performance of a CICS system

Important

Always tune DASD, the network, and the overall MVS system ***before*** tuning any individual CICS subsystem through CICS parameters.

Also review your application code before any further tuning

"Chapter 14. Performance checklists" on page 177 itemizes the actions you can take to tune the performance of an operational CICS system.

The other chapters in this part contain the relevant performance tuning guidelines for the following aspects of CICS:

Chapter 14. Performance checklists

The following checklists provide a quick reference to options that you can adjust to relieve different constraints. They assume that you have identified the exact cause of an existing constraint; they should ***not*** be used for random tuning exercises.

There are four checklists, corresponding to four of the main contention areas described in "Chapter 11. Identifying CICS constraints" on page 151.

1. I/O contention (this applies to data set and database subsystems, as well as to the data communications network)
2. Virtual storage above and below the 16MB line
3. Real storage
4. Processor cycles.

The checklists are in the sequence of low-level to high-level resources, and the items are ordered from those that probably have the greatest effect on performance to those that have a lesser effect, from the highest likelihood of being a factor in a normal system to the lowest, and from the easiest to the most difficult to implement.

Before taking action on a particular item, you should review the item to:

- Determine whether the item is applicable in your particular environment
- Understand the nature of the change
- Identify the trade-offs involved in the change.

Input/output contention checklist

Note:
Ideally, I/O contention should be reduced by using very large data buffers and keeping programs in storage. This would require adequate central and expanded storage, and programs that can be loaded above the 16MB line

Item	Page
VSAM considerations	
Review use of LLA	192
Implement Hiperspace buffers	235
Review/increase data set buffer allocations within LSR	230
Use data tables when appropriate	239
Database considerations	
Replace DL/I function shipping with IMS/ESA DBCTL facility	261
Reduce/replace shared database access to online data sets	261
Review DB2 threads and buffers	264
Journaling	

Item	Page
Increase activity keypoint frequency (AKPFREQ) value	277
Terminals, VTAM and SNA.	
Implement terminal output compression exit	210
Increase concurrent VTAM inputs	200
Increase concurrent VTAM logon/logoffs	206
Minimize SNA terminal data flows	203
Reduce SNA chaining	205
Miscellaneous	
Reduce DFHRPL library contention	296
Review temporary storage strings	317
Review transient data strings	322

Virtual storage above and below 16MB line checklist

Note:
The lower the number of concurrent transactions in the system, the lower the usage of virtual storage. Therefore, improving transaction internal response time decreases virtual storage usage. Keeping programs in storage above the 16MB line, and minimizing physical I/Os makes the largest contribution to well-designed transaction internal response time improvement.

Item	Page
CICS region	
Increase CICS region size	187
Reorganize program layout within region	296
Split the CICS region	281
DSA sizes	
Specify optimal size of the dynamic storage areas upper limits (DSALIM, EDSALIM)	620
Adjust maximum tasks (MXT)	285
Control certain tasks by transaction class	286
Put application programs above 16MB line	297
Database considerations	
Increase use of DBCTL and reduce use of shared database facility	261
Replace DL/I function shipping with IMS DBCTL facility	261
Review use of DB2 threads and buffers	264
Applications	
Compile COBOL programs RES, NODYNAM	312
Use PL/I shared library facility	313
Implement VS COBOL II	313
Journaling	

Item	Page
Increase activity keypoint frequency (AKPFREQ) value	277
Terminals, VTAM and SNA	
Reduce VTAM input message size	199
Reduce concurrent VTAM inputs	200
Reduce terminal scan delay	207
Discourage use of MSGINTEG and PROTECT	203
Reduce concurrent VTAM logon/logoffs	206
Reduce AIQMAX setting for autoinstall	211
MRO/ISC considerations	
Implement MVS cross-memory services with MRO	301
Implement MVS cross-memory services with shared database programs	301
Miscellaneous	
Reduce use of aligned maps	295
Prioritize transactions	289
Use only required CICS recovery facilities	329
Recycle job initiators with each CICS startup	189

Real storage checklist

Note:
Adequate central and expanded storage is vital to achieving good performance with CICS.

Item	Page
MVS considerations	
Dedicate, or fence, real storage to CICS	186
Make CICS nonswappable	186
Move CICS code to the LPA/ELPA	294
VSAM considerations	
Review the use of Hiperspace buffers	236
Use VSAM LSR where possible	235
Review the number of VSAM buffers	230
Review the number of VSAM strings	232
Task control considerations	
Adjust maximum tasks (MXT)	285
Control certain tasks by transaction class	286
MRO/ISC considerations	
Implement MVS cross-memory services with MRO	301
Implement MVS cross-memory services with shared database programs	
Use CICS intercommunication facilities	301
Database considerations	

Item	Page
Replace DL/I function shipping with IMS DBCTL facility	261
Review use of DB2 buffers and threads	264
Temporary storage and transient data	
Reduce temporary storage strings or buffers	317
Reduce transient data strings or buffers	322
Journaling	
Increase activity keypoint frequency (AKPFREQ) value	277
Terminal, VTAM and SNA	
Reduce terminal scan delay	207
Reduce concurrent VTAM inputs	200
Reduce VTAM input message size	199
Prioritize transactions	289
Reduce concurrent VTAM logon/logoffs	206
Applications	
Use PL/I shared library facilities	313
Compile COBOL programs RES, NODYNAM	312
Miscellaneous	
Decrease region exit interval	190
Reduce trace table size	328
Use only required CICS recovery facilities	329

Processor cycles checklist

Note:
Minimizing physical I/Os by employing large data buffers and keeping programs in storage reduces processor use, if adequate central and expanded storage is available.

Item	Page
General	
Reduce or turn off CICS trace	328
Increase CICS dispatching level or performance group	188
Terminal, VTAM and SNA	
Implement VTAM high performance option processing	202
Increase terminal scan delay	207
Minimize SNA terminal data flows	203
Reduce SNA chaining	205
Task control considerations	
Adjust maximum tasks (MXT)	285

Item	Page
Control certain tasks by transaction class	286
Define CICS maps with device suffixes	311
MRO/ISC considerations	
Implement MVS cross-memory services with MRO	301
Implement MRO fastpath facilities	301
Implement MVS cross-memory services with shared database programs	261
Use CICS intercommunication facilities	301
Database considerations	
Journaling	
Increase activity keypoint frequency (AKPFREQ) value	277
Temporary storage and transient data	
Increase temporary storage queue pointer allocations	317
Increase use of main temporary storage	317
Review the use of CICS transient data facilities	322
Miscellaneous	
Use only required CICS monitoring facilities	327
Review use of required CICS recovery facilities	329
Review use of required CICS security facilities	330
Increase region exit interval	190
Review use of program storage	296
Use NPDELAY for unsolicited input errors on TCAM lines	210
Prioritize transactions	289

Chapter 15. MVS and DASD

The information in this chapter appears under the following headings:

Tuning CICS and MVS

Tuning CICS for virtual storage under MVS depends on the following main elements:

- MVS systems tuning
- VTAM tuning
- CICS tuning
- VSAM tuning.

Because tuning is a top-down activity, you should already have made a vigorous effort to tune MVS before tuning CICS. Your main effort to reduce virtual storage constraint and to get relief should be concentrated on reducing the life of the various individual transactions: in other words, shortening task life.

This section describes some of the techniques that can contribute significantly to shorter task life, and therefore, a reduction in virtual storage constraint.

The installation of a faster processor can cause the current instructions to be executed faster and, therefore, reduce task life (internal response time), because more transactions can be processed in the same period of time. Installing faster DASD can reduce the time spent waiting for I/O completion, and this shorter wait time for paging operations, data set index retrieval, or data set buffer retrieval can also reduce task life in the processor.

Additional real storage, if page-ins are frequently occurring (if there are more than 5 to 10 page-ins per second, CICS performance is affected), can reduce waits for the paging subsystem.

MVS provides storage isolation for an MVS performance group, which allows you to reserve a specific range of real storage for the CICS address space and to control the page-rates for that address space based on the task control block (TCB) time absorbed by the CICS address space during execution.

You can isolate CICS data on DASD drives, strings, and channels to minimize the I/O contention suffered by CICS from other DASD activity in the system. Few

CICS online systems generate enough I/O activity to affect the performance of CICS seriously if DASD is isolated in this manner.

So far (except when describing storage isolation and DASD sharing), we have concentrated on CICS systems that run a stand-alone single CICS address space. The sizes of all MVS address spaces are defined by the common requirements of the largest subsystem. If you want to combine the workload from two or more processors onto an MVS image, you must be aware of the virtual storage requirements of each of the subsystems that are to execute on the single-image ESA processor. Review the virtual storage effects of combining the following kinds of workload on a single-image MVS system:

1. CICS and a large number (100 or more) of TSO users
2. CICS and a large IMS system
3. CICS and 5000 to 7500 VTAM LUs.

By its nature, CICS requires a large private region that may not be available when the large system's common requirements of these other subsystems are satisfied. If, after tuning the operating system, VTAM, VSAM, and CICS, you find that your address space requirements still exceed that available, you can split CICS using one of three options:

1. Multiregion option (MRO)
2. Intersystem communication (ISC)
3. Multiple independent address spaces.

Adding large new applications or making major increases in the size of your VTAM network places large demands on virtual storage, and you must analyze them before implementing them in a production system. Careful analysis and system specification can avoid performance problems arising from the addition of new applications in a virtual-storage-constrained environment.

If you have not made the necessary preparations, you usually become aware of problems associated with severe stress only after you have attempted to implement the large application or major change in your production system. Some of these symptoms are:

- Poor response times
- Short-on-storage
- Program compression
- Heavy paging activity
- Many well-tested applications suddenly abending with new symptoms
- S80A and S40D abends
- S822 abends
- Dramatic increase in I/O activity on DFHRPL program libraries.

Various chapters in the rest of this book deal with specific, individual operands and techniques to overcome these problems. They tell you how to minimize the use of virtual storage in the CICS address space, and how to split it into multiple address spaces if your situation requires it.

For an overall description of ESA virtual storage, see "Appendix F. MVS and CICS virtual storage" on page 611.

Reducing MVS common system area requirements

This can be the most productive area for tuning. CICS installations that have not previously tuned their ESA system may be able to recover 1.5 to 2.0 megabytes of virtual storage. This topic is outside the scope of this book, but you should investigate it fully before tuning CICS. A manual that gives information about this is the *OS/390 MVS Initialization and Tuning Reference* manual.

Splitting online systems: availability

Splitting the CICS system into two or more separate address spaces may lead to improved availability. If CICS failures are being caused by application program errors, for example, separating out the failing application can improve overall availability. This can also give virtual storage gains and, in addition, can allow you to use multiprocessors and MVS images more efficiently. See "Splitting online systems: virtual storage" on page 281 for more information. A fuller account can be found in the *System/390 MVS Sysplex Application Migration* Guide (*GC28-1211*).

The availability of the overall system may be improved by splitting the system because the effects of a failure can be limited or the time to recover from the failure can be reduced.

The main ways of splitting a system for availability are to have:

- *Terminal owning regions*. With one or more terminal owning regions (TORs) using transaction routing, availability can be improved because a TOR is less likely to fail because it contains no application code. The time taken to restart the failed part of the system is reduced because the terminal sessions are maintained at failure if the TOR continues to operate.
- *Multiple application owning regions*. Using multiple application owning regions (AORs), you can separate unstable or new applications from the rest of the system. If these applications cause a failure of that AOR, all other AORs are still available. If the region susceptible to failure contains no terminals or files and databases, it also tends to restart quickly.

  Applications under test in AORs can use function shipping to access 'live' data, which adds to the realism of the test environment.
- *File owning regions*. File requests from many CICS regions can be function-shipped to file owning regions (FORs). The FORs contain no application code and so are unlikely to fail, so that access to files can be maintained even if other regions fail. Removing the files and databases from these other regions speeds up their recovery by removing file allocation and opening time.

Having only one FOR in a system, or logical subset of a system, can reduce the operational difficulties of restarting a system. It is possible to split the regions in different ways to those described so far, by having many regions all of which own some terminals, some applications, and some files and databases. This type of splitting is very complex to maintain and operate, and also needs careful monitoring to ensure that the performance of the overall system is optimal. For these reasons, a structured approach with each of the regions having a clearly defined set of one type of resource is recommended.

Limitations

Splitting a CICS system requires increased real storage, increased processor cycles, and extensive planning. These overheads are described in more detail in "Splitting online systems: virtual storage" on page 281.

Recommendations

If availability of your system is an important requirement, both splitting systems and the use of XRF should be considered. The use of XRF can complement the splitting of systems by automating the recovery of the components.

When splitting your system, you should try to separate the sources of failure so that as much of the rest of the system as possible is protected against their failure, and remains available for use. Critical components should be backed up, or configured so that service can be restored with minimum delay. Since the advantages of splitting regions for availability can be compromised if the queueing of requests for remote regions is not controlled, you should also review "Intersystems session queue management" on page 303.

Making CICS nonswappable

You can take a variety of actions to cause the operating system to give CICS preferential treatment in allocation of processor resources.

Making CICS nonswappable prevents the address space from being swapped out in MVS, and reduces the paging overhead. Consider leaving only very lightly used test systems swappable.

How implemented

You should consider making your CICS region nonswappable by using the PPTNSWP option in the MVS Program Properties Table (PPT).

Limitations

Using the PPT will make all CICS systems (including test systems) nonswappable. As an alternative, use the IPS. For more information about defining entries in the PPT see the *OS/390 MVS Programming: Callable Services for HLL* manual.

How monitored

The DISPLAY ACTIVE (DA) command on SDSF gives you an indication of the number of real pages used and the paging rate. Use RMF, the RMFMON command on TSO to provide additional information. For more information about RMF see "Resource measurement facility (RMF)" on page 25 or the *MVS RMF User's Guide*.

Isolating (fencing) real storage for CICS (PWSS and PPGRTR)

Real storage isolation, or "fencing" in MVS, is a way of allocating real storage to CICS alone, and can reduce paging problems.

Recommendations

Use PWSS=(a,b) and PPGRTR=(c,d) or PPGRT=(c,d) in the IEAIPSxx.

The PWSS=(a,b) parameter specifies the range (minimum,maximum) of page frames needed for the target working set size for an address space.

The target working set size of an XRF alternate CICS system can vary significantly in different environments.

The PPGRTR=(c,d) or PPGRT=(c,d) parameter specifies the minimum and maximum paging rates to use in adjusting the target working set size specified in

PWSS. PPGRTR means that the system resource manager (SRM) calculates the paging rate using the alternate system's residency time, rather than the execution time if PPGRT is specified.

For the XRF alternate system that has a low activity while in the surveillance phase, PPGRTR is a better choice because the target working set size is adjusted on the basis of page-faults per second, rather than page-faults per execution second.

During catchup and while tracking, the real storage needs of the XRF alternate CICS system are increased as it changes terminal session states and the contents of the TCT. At takeover, the real storage needs also increase as the alternate CICS system begins to switch terminal sessions and implement emergency restart. In order to ensure good performance and minimize takeover time, the target working set size should be increased. This can be done in several different ways, two of which are:

1. Parameter "b" in PWSS=(a,b) can be set to "*" which allows the working set size to increase without limit, if the maximum paging rate (parameter "d" in PPGRTR=(c,d)) is exceeded.
2. A command can be put in the CLT to change the alternate CICS system's performance group at takeover to one which has different real storage isolation parameters specified.

If you set PWSS=(*,*), and PPGRTR=(1,2), this allows CICS to use as much storage as it wants when the paging rate is > 2 per second. The values depend very much on the installation and the MVS setup. The values suggested here assume that CICS is an important address space and therefore needs service to be resumed quickly.

For the definition and format of the storage isolation parameters in IEAIPSxx, see the *OS/390 MVS Initialization and Tuning Reference* manual.

How implemented

See the *OS/390 MVS Initialization and Tuning Reference* manual.

How monitored

Use RMF, the RMFMON command on TSO for additional information. The DISPLAY ACTIVE (DA) command on SDSF will give you an indication of the number of real pages used and the paging rate.

Increasing the CICS region size

If all other factors in a CICS system are kept constant, increasing the region size available to CICS allows an increase in the dynamic storage areas.

Changes to MVS and other subsystems over time generally reduce the amount of storage required below the 16MB line. Thus the CICS region size may be able to be increased when a new release of MVS or non-CICS subsystem is installed.

To get any further increase, operating-system functions and storage areas (such as the local shared queue area, LSQA), or other programs must be reduced. The LSQA is used by VTAM and other programs, and any increase in the CICS region size decreases the area available for the LSQA, SWA, and subpools 229 and 230. A shortage in these subpools can cause S80A, S40D, and S822 abends.

If you specify a larger region, the value of the relevant dsasize system initialization parameter must be increased or the extra space is not used.

How implemented

The region size is defined in the startup job stream for CICS. Other definitions are made to the operating system or through operating-system console commands.

To determine the maximum region size, determine the size of your private area from RMF II or one of the storage monitors available.

To determine the maximum region size you should allocate, use the following formula:

Max region possible = private area size – system region size – (LSQA + SWA + subpools 229 and 230)

The remaining storage is available for the CICS region; for safety, use 80% or 90% of this number. If the system is static or does not change much, use 90% of this number for the REGION= parameter; if the system is dynamic, or changes frequently, 80% would be more desirable.

Note: You must maintain a minimum of 200KB of free storage between the top of the region and the bottom of the ESA high private area (the LSQA, the SWA, and subpools 229 and 230).

How monitored

Use RMF, the RMFMON command on TSO for additional information. For more information about RMF see "Resource measurement facility (RMF)" on page 25 or the *MVS RMF User's Guide*.

Giving CICS a high dispatching priority or performance group

Giving CICS a high dispatching priority causes the processor to be accessible more often when it is needed.

Performance groups in MVS are another way of giving CICS increased access to the processor. Putting CICS at a high dispatching priority or in a favorable performance group is most effective when CICS is processor-constrained.

The relative order of priority can be:

- VTAM
- Performance monitor
- Database
- CICS.

How implemented

Set the CICS priority above the automatic priority group (APG). See the *OS/390 MVS Initialization and Tuning Reference* manual for further information.

There are various ways to assign CICS a dispatching priority. The best is through the ICS (PARMLIB member IEAICSxx). The ICS assigns performance group numbers and enforces assignments. The dispatching priorities are specified in PARMLIB member IEAIPSxx. Use APGRNG to capture the top ten priority sets (6

through 15). Specify a suitably high priority for CICS. There are priority levels that change dynamically, but we recommend a simple fixed priority for CICS. Use storage isolation only when necessary.

You cannot specify a response time, and you must give CICS enough resources to achieve good performance.

See the *OS/390 MVS Initialization and Tuning Reference* manual for more information.

How monitored

Use either the DISPLAY ACTIVE (DA) command on SDSF or use RMF, the RMFMON command on TSO. For more information about RMF see "Resource measurement facility (RMF)" on page 25 or the *MVS RMF User's Guide*.

Using job initiators

The management of the MVS high private area can sometimes result in fragmentation and stranded subpools caused by large imbedded free areas known as "holes".

Some fragmentation can also occur in a region when a job initiator starts multiple jobs without being stopped and then started again. If you define the region as having the maximum allowable storage size, it is possible to start and stop the job the first time the initiator is used, but to have an S822 abend (insufficient virtual storage) the second time the job is started. This is because of the fragmentation that occurs.

In this situation, either the region has to be decreased, or the job initiator has to be stopped and restarted.

Two methods of starting the CICS job are available, to maximize the virtual storage available to the region. One is to start and stop the initiator with each initialization of CICS, executing CICS in a newly started initiator; and the other is to use the MVS START command.

If CICS is executed as an MVS-started task (using the MVS START command) instead of submitting it as a batch job, this not only ensures that a clean address space is used (reducing the possibility of an S822 abend), but also saves a significant amount of LSQA storage.

Effects

Some installations have had S822 abends after doing I/O generations or after adding DD statements to large applications. An S822 abend occurs when you request a REGION=nnnnK size that is larger than the amount available in the address space.

The maximum region size that is available is difficult to define, and is usually determined by trial and error. One of the reasons is that the size depends on the system generation and on DD statements.

At least two techniques can be used to reduce storage fragmentation:

1. *Dynamic allocation*. You might consider writing a "front-end" program that dynamically allocates the cataloged data sets for the step and then transfers control (XCTL) to CICS. The effect of this is that only one eligible device list (EDL) is used at a time.
2. *UNITNAME*. You might consider creating a new UNITNAME (via EDT-GEN or IOGEN). This UNITNAME could be a subset of devices known to contain the cataloged data set. By using the "unit override" feature of JCL, it could cause the EDL to be limited to the devices specified in the UNITNAME.

Limitations

Available virtual storage is increased by starting new initiators to run CICS, or by using MVS START. Startup time may be minimally increased.

How implemented

CICS startup and use of initiators are defined in an installation's startup procedures.

How monitored

Part of the job termination message IEF374I 'VIRT=nnnnnK' shows you the virtual storage below the 16MB line, and another part 'EXT=nnnnnnnK' shows the virtual storage above the 16MB line.

Region exit interval (ICV)

When CICS cannot dispatch a task, either because there are no tasks in the system at that time, or because all tasks are waiting for data set or terminal I/O to finish, CICS issues an operating-system WAIT. The ICV system initialization parameter (see also "Terminal scan delay (ICVTSD)" on page 207) controls the length of this wait (but bear in mind that any interrupt, for example, data set I/O or terminal I/O, before any of these expires, causes CICS to be dispatched).

The ICV system initialization parameter specifies the maximum time in milliseconds that CICS releases control to the operating system when there are no transactions ready to resume processing. CICS issues a region wait in this case for the time specified in the ICV system initialization parameter. If activity in the system causes CICS to be dispatched sooner, this parameter has no effect.

In general, ICV can be used in low-volume systems to keep part of the CICS management code paged in. Expiration of this interval results in a full terminal control table (TCT) scan in non-VTAM environments, and controls the dispatching of terminal control in VTAM systems with low activity. Redispatch of CICS by MVS after the wait may be delayed because of activity in the supervisor or in higher-priority regions, for example, VTAM. The ICV delay can affect the shutdown time if no other activity is taking place.

The value of ICV acts as a backstop for MROBTCH (see "Batching requests (MROBTCH)" on page 306).

Main effect

The region exit interval determines the maximum period between terminal control full scans. However, the interval between full scans in very active systems may be less than this, being controlled by the normally shorter terminal scan delay interval (see "Terminal scan delay (ICVTSD)" on page 207). In such systems, ICV becomes largely irrelevant unless ICVTSD has been set to zero.

Secondary effects

Whenever control returns to the task dispatcher from terminal control after a full scan, ICV is added to the current time of day to give the provisional due time for the next full scan. In idle systems, CICS then goes into an operating-system wait state, setting the timer to expire at this time. If there are application tasks to dispatch, however, CICS passes control to these and, if the due time arrives before CICS has issued an operating-system WAIT, the scan is done as soon as the task dispatcher next regains control.

In active systems, after the due time has been calculated by adding ICV, the scan may be performed at an earlier time by application activity (see "Terminal scan delay (ICVTSD)" on page 207).

Operating-system waits are not always for the duration of one ICV. They last only until some event ends. One possible event is the expiry of a time interval, but often CICS regains control because of the completion of an I/O operation. Before issuing the operating-system WAIT macro, CICS sets an operating-system timer, specifying the interval as the time remaining until the next time-dependent activity becomes due for processing. This is usually the next terminal control scan, controlled by either ICV or ICVTSD, but it can be the earliest ICE expiry time, or even less.

In high-activity systems, where CICS is contending for processor time with very active higher-priority subsystems (VTAM, TSO, other CICS systems, or DB/DC), control may be seized from CICS so often that CICS always has work to do and never issues an operating-system WAIT.

Where useful

The region exit interval is useful in environments where batch or other CICS systems are running concurrently.

Limitations

Too low a value can impair concurrent batch performance by causing frequent and unnecessary dispatches of CICS by MVS. Too high a value can lead to an appreciable delay before the system handles time-dependent events (such as abends for terminal read or deadlock timeouts) after the due time.

A low ICV value does not prevent all CICS modules from being paged out. When the ICV time interval expires, the operating system dispatches CICS task control which, in turn, dispatches terminal control. CICS references only task control, terminal control, TCT, and the CSA. No other modules in CICS are referenced. If there is storage constraint they do not stay in real storage.

After the operating-system WAIT, redispatch of CICS may be delayed because of activity in the supervisor or in higher-priority regions such as VTAM, and so on.

The ICV delay can affect the shutdown time if no other activity is taking place.

Recommendations

The time interval can be any decimal value in the range from 100 through 3600000 milliseconds.

In normal systems, set ICV to 1000-10000 milliseconds, or more.

A low interval value can enable much of the CICS nucleus to be retained, and not be paged out at times of low terminal activity. This reduces the amount of paging necessary for CICS to process terminal transactions (thus representing a potential reduction in response time), sometimes at the expense of concurrent batch region throughput. Large networks with high terminal activity tend to drive CICS without a need for this value, except to handle the occasional, but unpredictable, period of inactivity. These networks can usually function with a large interval (10000 to 30000 milliseconds). After a task has been initiated, the system recognizes its requests for terminal services and the completion of the services, and overrides this maximum delay interval.

Small systems or those with low terminal activity are subject to paging introduced by other jobs running in competition with CICS. If you specify a low interval value, key portions of the CICS nucleus are referenced more frequently, thus reducing the probability of these pages being paged-out. However, the execution of the logic, such as terminal polling activity, without performing productive work might be considered wasteful.

You must weigh the need to increase the probability of residency by frequent but unproductive referencing, against the extra overhead and longer response times incurred by allowing the paging to occur. If you increase the interval size, more productive work is performed at the expense of performance if paging occurs during the periods of CICS activity.

Note: If the terminal control negative poll delay feature is used, the ICV value selected must not exceed the negative poll delay value. If the negative poll delay used is zero, any ICV value may be used (see "Negative poll delay (NPDELAY)" on page 210).

How implemented

ICV is specified in the SIT or at startup, and can be changed using either the CEMT or EXEC CICS SET SYSTEM (time) command. It is defined in units of milliseconds, rounded down to the nearest multiple of ten. The default is 1000 (that is, one second; usually too low).

How monitored

The region exit interval can be monitored by the frequency of CICS operating-system WAITs that are counted in "Dispatcher domain" on page 360.

Use of LLA (MVS library lookaside)

Modules loaded by CICS from the DFHRPL libraries may be managed by the MVS LLA (library lookaside) facility. LLA is designed to minimize disk I/O by keeping load modules in a VLF (virtual lookaside facility) dataspace and keeping a version of the library directory in its own address space.

LLA manages modules (system or application) whose library names you have put in the appropriate CSVLLA member in SYS1.PARMLIB.

There are two optional parameters in this member that affect the management of specified libraries:

FREEZE
Tells the system always to use the copy of the directory that is maintained in the LLA address space.

NOFREEZE
Tells the system always to search the directory that resides in DASD storage.

However, FREEZE and NOFREEZE are only relevant when LLACOPY is not used. When CICS issues a LOAD and specifies the directory entry (DE), it bypasses the LLA directory processing, but determines from LLA whether the program is already in VLF or must be fetched from DASD. For more information about the FREEZE and NOFREEZE options, see the *OS/390 MVS Initialization and Tuning Guide*.

The use of LLA to manage a very busy DFHRPL library can show two distinct benefits:
1. Improved transaction response time
2. Better DASD utilization.

It is possible, as throughput increases, that DASD utilization actually decreases. This is due to LLA's observation of the load activity and its decisions about which modules to stage (keep) in the VLF dataspace.

LLA does not automatically stage all members that are fetched. LLA attempts to select those modules whose staging gives the best reductions in response time, contentions, storage cost, and an optional user-defined quantity.

In addition to any USER-defined CICS DFHRPL libraries, LLA also manages the system LNKLST. It is likely that staging some modules from the LNKLST could have more effect than staging modules from the CICS libraries. LLA makes decisions on what is staged to VLF only after observing the fetch activity in the system for a certain period. For this reason it is possible to see I/O against a program library even when it is managed by LLA.

Another contributing factor for continued I/O is the system becoming "MAXVIRT constrained", that is, the sum of bytes from the working set of modules is greater than the MAXVIRT parameter for the LLA class of VLF objects. You can increase this value by changing it in the COFVLF member in SYS1.PARMLIB. A value too small can cause excessive movement of that VLF object class; a value too large can cause excessive paging; both may increase the DASD activity significantly.

See the *OS/390 MVS Initialization and Tuning Guide* manual for information on LLA and VLF parameters.

Effects of LLACOPY

CICS can use one of two methods for locating modules in the DFHRPL concatenation. Either a build link-list (BLDL) macro or a LLACOPY macro is issued to return the directory information to pass to the load request. Which macro is issued is dependant upon the LLACOPY system initialization parameter and the reason for the locate of the module.

The LLACOPY macro is used to update the LLA-managed directory entry for a module or a list of modules. If a module which is LLA managed has an LLACOPY issued against it, it results in a BLDL with physical I/O against the DCB specified. If the directory information does not match that which is stored within LLA, the LLA tables are then updated, keeping both subsystems synchronized. While this activity takes place an ENQ for the resource SYSZLLA1.update is held. This is then

unavailable to any other LLACOPY request on the same MVS system and therefore another LLACOPY request is delayed until the ENQ is released.

The BLDL macro also returns the directory information. When a BLDL is issued against an LLA managed module, the information returned will be from the LLA copy of the directory, if one exists. It will not necessarily result in physical I/O to the dataset and may therefore be out of step with the actual dataset. BLDL does not require the SYSZLLA1.update ENQ and is therefore less prone to being delayed by BLDLs on the same MVS system. Note that it is not advisable to use a NOCONNECT option when invoking the BLDL macro because the DFHRPL concatenated dataset may contain partitioned data set extended (PDSE) datasets. PDSE can contain more function than PDS, but CICS may not recognise some of this function. PDSE also use more virtual storage .

The SIT Parameter LLACOPY

If you code LLACOPY=YES, the default, CICS issues a LLACOPY macro each time a module is located from the RPL dataset. This is done either on the first ACQUIRE or on any subsequent NEWCOPY or PHASEIN requests. This ensures that CICS always obtains the latest copy of any LLA-managed modules. There is a small chance of delay because of a failure to obtain an ENQ while another LLACOPY completes and there is some extra pathlength involved in maintaining the LLA tables.

If you code LLACOPY=NO, CICS never issues an LLACOPY macro. Instead, each time the RPL dataset is searched for a module, a BLDL is issued.

If you code LLACOPY=NEWCOPY to request a new copy of an LLA-managed module, a MODIFY LLA,REFRESH or F LLA,REFRESH must be issued before the NEWCOPY is performed within CICS. (MODIFY LLA,REFRESH rebuilds LLA's directory for the entire set of libraries managed by LLA.) When you code LLACOPY=NEWCOPY, CICS issues the LLACOPY macro when loading a module as a result of a NEWCOPY or PHASEIN request. A BLDL macro is issued on the first use of the module, but when it is due to one of these requests, the REFRESH option results in loading the new module. If an out of date version of a module is loaded upon its first use, the latest version would be used after a NEWCOPY or PHASEIN. For more information about the LLACOPY system initialization parameter, see the *CICS System Definition Guide*.

DASD tuning

The main solutions to DASD problems are to:

- Reduce the number of I/O operations
- Tune the remaining I/O operations
- Balance the I/O operations load.

Reducing the number of I/O operations

The principal ways of reducing the number of I/O operations are to:

- Allocate VSAM Hiperspace buffers
- Allocate additional address space buffers
- Use data tables when appropriate
- Use or increase the use of main temporary storage
- Eliminate or minimize program compression
- Review and improve the design of applications run on CICS

- Make use of a DASD controller cache, but only if data set placement tuning has been done
- Minimize CI/CA splits by:
  - Allocating ample free space (free space can be altered by key range during load)
  - Timely reorganizations of disk storage.

Tuning the I/O operations

This can reduce service time. The principal ways of tuning the I/O operations are to:

- Specify the correct CI size. This has an effect on:
  - The space used on the volume
  - Transfer time
  - Storage requirements for buffers
  - The type of processing (direct or sequential).
- Specify the location of the VTOC correctly.
- Take care over data set placement within the volume.
- Use an appropriately fast device type and, if necessary, use a cache memory (but only if data set placement tuning has been done and if there are sufficient channels to handle the device speed).

Balancing I/O operations

This can reduce queue time. The principal ways of balancing I/O operations are to:

- Spread a high-use data set across multiple volumes.
- Minimize the use of shared DASD volumes between multiple processors.
- Place batch files and online files on separate volumes, especially:
  - Spool files
  - Sort files
  - Assembler or compiler work files
  - Page data sets.
- Place index and data on separate volumes (for VSAM KSDS files).
- Place concurrently used files on separate volumes. For example, a CICS journal should be the only data set in use on its volume.

Take the following figures as guidelines for best DASD response times for online systems:

- Channel busy: less than 30% (with CHP ids this can be higher)
- Device busy: less than 35% for randomly accessed files
- Average response time: less than 20 milliseconds.

Aim for multiple paths to disk controllers because this allows dynamic path selection to work.

Chapter 16. Networking and VTAM

This chapter includes the following topics:

Terminal input/output area (TYPETERM IOAREALEN or TCT TIOAL)

If you are using VTAM, the CEDA DEFINE TYPETERM IOAREALEN command determines the initial size of the terminal input/output area (TIOA) to be passed onto a transaction for each terminal. The syntax for IOAREALEN is ({0 | value1},{0 | value2}). This operand is used only for the first input message for all transactions.

For TCAM, the DFHTCT TYPE=TERMINAL TIOAL=value macro, is the only way to adjust this value.

One value defining the minimum size is used for non-SNA devices, while two values specifying both the minimum and maximum size are used for SNA devices.

This book does not discuss the performance aspects of the CICS Front End Programming Interface. See the *CICS Front End Programming Interface User's Guide* for more information.

Effects

When value1,0 is specified for IOAREALEN, value1 is the minimum size of the terminal input/output area that is passed to an application program when a RECEIVE command is issued. If the size of the input message exceeds value1, the area passed to the application program is the size of the input message.

When value1, value2 is specified, value1 is the minimum size of the terminal input/output area that is passed to an application program when a RECEIVE command is issued. Whenever the size of the input message exceeds value1, CICS will use value2. If the input message size exceeds value2, the node abnormal condition program sends an exception response to the terminal.

If you specify ATI(YES), you must specify an IOAREALEN of at least one byte.

For TCAM supported devices, if the TIOAL operand is omitted, it defaults to the INAREAL length value in the TCT TYPE=LINE operand. Do not omit the TIOAL operand for remote terminals.

Limitations

Real storage can be wasted if the IOAREALEN (value1) or TIOAL value is too large for most terminal inputs in the network. If IOAREALEN (value1) or TIOAL is smaller than most initial terminal inputs, excessive GETMAIN requests can occur, resulting in additional processor requirements, unless IOAREALEN(value1) or TIOAL is zero.

Recommendations

IOAREALEN(value1) or TIOAL should be set to a value that is slightly larger than the average input message length for the terminal. The maximum value that may be specified for IOAREALEN/TIOAL is 32767 bytes.

If a value of nonzero is required, the best size to specify is the most commonly encountered input message size. A multiple of 64 bytes minus 21 allows for SAA requirements and ensures good use of operating system pages.

For VTAM, you can specify two values if inbound chaining is used. The first value should be the length of the normal chain size for the terminal, and the second value should be the maximum size of the chain. The length of the TIOA presented to the task depends on the message length and the size specified for the TIOA. (See the example in Figure 31.)

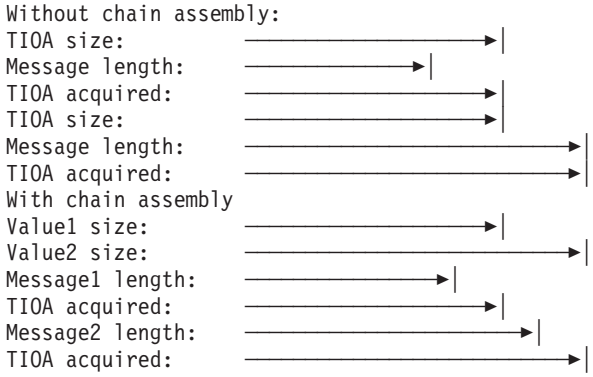


Figure 31. Message length and terminal input/output area length

Avoid specifying too large a value1, for example, by matching it to the size of the terminal display screen. This area is used only as input. If READ with SET is specified, the same pointer is used by applications for an output area.

If too small a value is specified for value1, extra processing time is required for chain assembly, or data is lost if inbound chaining is not used.

In general, a value of zero is best because it causes the optimum use of storage and eliminates the second GETMAIN request. If automatic transaction initiation (ATI) is used for that terminal, a minimum size of one byte is required.

The second value for SNA devices is used to prevent terminal streaming, and so should be slightly larger than the largest possible terminal input in the network. If

a message larger than this second value is encountered, a negative response is returned to the terminal, and the terminal message is discarded.

How implemented

For VTAM, the TIOA value is specified in the CEDA DEFINE TYPETERM IOAREALEN attribute.

For TCAM, the TIOAL value can be specified in the terminal control table (TCT) TYPE=TERMINAL operand. TIOAL defaults to the INAREAL value specified in the TCT TYPE=LINE operand.

How monitored

RMF and NetView Performance Monitor (NPM) can be used to show storage usage and message size characteristics in the network.

Receive-any input areas (RAMAX)

The system initialization parameter, RAMAX, specifies the size in bytes of the I/O area that is to be allocated for each VTAM receive-any operation. These storage areas are called receive-any input areas (RAIAs), and are used to receive the first terminal input for a transaction from VTAM. All input from VTAM comes in request/response units (RUs).

Storage for the RAIAs, which is above the 16MB line, is allocated by the CICS terminal control program during CICS initialization, and remains allocated for the entire execution of the CICS job step. The size of this storage is the product of the RAPOOL and RAMAX system initialization parameters.

Effects

VTAM attempts to put any incoming RU into the initial receive-any input area, which has the size of RAMAX. If this is not large enough, VTAM indicates that and also states how many extra bytes are waiting that cannot be accommodated.

RAMAX is the largest size of any RU that CICS can take directly in the receive-any command, and is a limit against which CICS compares VTAM's indication of the overall size of the RU. If there is more, VTAM saves it, and CICS gets the rest in a second request.

With a small RAMAX, you reduce the virtual storage taken up in RAIAs but risk more processor usage in VTAM retries to get any data that could not fit into the RAIA.

For many purposes, the default RAMAX value of 256 bytes is adequate. If you know that many incoming RUs are larger than this, you can always increase RAMAX to suit your system.

For individual terminals, there are separate parameters that determine how large an RU is going to be from that device. It makes sense for RAMAX to be at least as large as the largest CEDA SENDSIZE for any frequently-used terminals.

Where useful

You can use the RAMAX system initialization parameter in any networks that use the VTAM access method for terminals.

Limitations

Real storage can be wasted with a high RAMAX value, and additional processor time can be required with a low RAMAX value. If the RAMAX value is set too low, extra processor time is needed to acquire additional buffers to receive the remaining data. Because most inputs are 256 bytes, this should normally be specified.

Do not specify a RAMAX value that is less than the RUSIZE (from the CINIT) for a pipeline terminal because pipelines cannot handle overlength data.

Recommendations

Code RAMAX with the size in bytes of the I/O area allocated for each receive-any request issued by CICS. The maximum value is 32767.

Set RAMAX to be slightly larger than your CICS system input messages. If you know the message length distribution for your system, set the value to accommodate the majority of your input messages.

In any case, the size required for RAMAX need only take into account the first (or only) RU of a message. Thus, messages sent using SNA chaining do not require RAMAX based on their overall chain length, but only on the size of the constituent RUs.

Receive-any input areas are taken from a fixed length subpool of storage. A size of 2048 may appear to be adequate for two such areas to fit on one 4KB page, but only 4048 bytes are available in each page, so only one area fits on one page. A size of 2024 should be defined to ensure that two areas, including page headers, fit on one page.

How implemented

RAMAX is a system initialization parameter.

How monitored

The size of RUs or chains in a network can be identified with a VTAM line or buffer trace. The maximum size RUs are defined in the CEDA SENDSIZE attribute.

Receive-any pool (RAPOOL)

The RAPOOL system initialization parameter specifies the number of concurrent receive-any requests that CICS is to process from VTAM. RAPOOL determines how many receive-any buffers there are at any time and, therefore, if VTAM has a lot of input simultaneously, it enables VTAM to put all the messages directly into CICS buffers rather than possibly having to store them itself elsewhere. The first operand (value1) is for non-HPO systems, the second operand (value2) is for HPO systems.

The HPO value for the non-HPO operand is derived according to the formula shown in the *CICS System Definition Guide*. The second operand (value2) for HPO systems is used with minimal adjustment by the formula.

Effects

Initially, task input from a terminal or session is received by the VTAM access method and is passed to CICS if CICS has a receive-any request outstanding.

For each receive-any request, a VTAM request parameter list (RPL), a receive-any control element (RACE), and a receive-any input area (RAIA)—the value specified by RAMAX (see "Receive-any input areas (RAMAX)" on page 199) are set aside. The total area set aside for VTAM receive-any operations is:

(maximum RAIA size + RACE size + RPL size) * RAPOOL

If HPO=YES, both RACE and RPL are above the 16MB line.

See page 199 for RAIA considerations.

In general, input messages up to the value specified in RAPOOL are all processed in one dispatch of the terminal control task. Because the processing of a receive-any request is a short operation, at times more messages than the RAPOOL value may be processed in one dispatch of terminal control. This happens when a receive-any request completes before the terminal control program has finished processing and there are additional messages from VTAM.

VTAM receive-any processing is for the first terminal message in a transaction, so RAPOOL has no effect on further inputs for conversational tasks. Those additional inputs are processed with VTAM receive-specific requests.

The pool is used only for the first input to start a task; it is not used for output or conversational input. VTAM posts the event control block (ECB) associated with the receive any input area. CICS then moves the data to the terminal I/O area (TIOA) ready for task processing. The RAIA is then available for reuse.

Where useful

Use the RAPOOL operand in networks that use the VTAM access method for terminals.

Limitations

If the RAPOOL value is set too low, this can result in terminal messages not being processed in the earliest dispatch of the terminal control program, thereby inducing transaction delays during high-activity periods. For example, if you use the default and five terminal entries want to start up tasks, three tasks may be delayed for at least the time required to complete the VTAM receive-any request and copy the data and RPL. In general, no more than 5 to 10% of all receive-any processing should be at the RAPOOL ceiling, with none being at the RAPOOL ceiling if there is sufficient storage.

If the RAPOOL value is set too high, this can use excessive virtual storage, but does not affect real storage because the storage is not page-fixed and is therefore paged out.

Recommendations

Whether RAPOOL is significant or not depends on the environment of the CICS system: whether, for example, HPO is being used.

In some cases, it may sometimes be more economical for VTAM to store the occasional peak of messages in its own areas rather than for CICS itself to have a large number of RAIAs, many of which are unused most of the time.

Furthermore, there are situations where CICS reissues a receive-any as soon as it finds one satisfied. It thereby uses the same element over and over again in order to bring in any extra messages that are in VTAM.

CICS maintains a VTAM RECEIVE ANY for n of the RPLs, where n is either the RAPOOL value, or the MXT value minus the number of currently active tasks, whichever is the smaller. See the *CICS System Definition Guide* for more information about these SIT parameters.

A general recommendation is to code RAPOOL with the number of fixed request parameter lists (RPLs) that you require. When it is not at MXT, CICS maintains a receive-any request for each of these RPLs. The number of RPLs that you require depends on the expected activity of the system, the average transaction lifetime, and the MXT specified.

The RAPOOL value you set depends on the number of sessions, the number of terminals, and the ICVTSD value (see page 207) in the system initialization table (SIT). Initially, for non-HPO systems, you should set RAPOOL to 1.5 times your peak *local* [2] transaction rate per second plus the autoinstall rate. This can then be adjusted by analyzing the CICS VTAM statistics and by resetting the value to the maximum RPLs reached.

For HPO systems, a small value (<= 5) is usually sufficient if specified through the value2 in the RAPOOL system initialization parameter. Thus, RAPOOL=20, for example, is specified either RAPOOL=(20) or RAPOOL=(20,5) to achieve the same effect.

How implemented

RAPOOL is a system initialization parameter.

How monitored

The CICS VTAM statistics contain values for the maximum number of RPLs posted on any one dispatch of the terminal control program, and the number of times the RPL maximum was reached. This maximum value may be greater than the RAPOOL value if the terminal control program is able to reuse an RPL during one dispatch. See "VTAM statistics" on page 47 for more information.

High performance option (HPO) with VTAM

The MVS high performance option (HPO) can be used for processing VTAM requests. The purpose of HPO is to reduce the transaction pathlength through VTAM.

Effects

HPO bypasses some of the validating functions performed by MVS on I/O operations, and implements service request block (SRB) scheduling. This shortens the instruction pathlength and allows some concurrent processing on MVS images for the VTAM operations because of the SRB scheduling. This makes it useful in a multi processor environment, but not in a single processor environment.

2. The RAPOOL figure does not include MRO sessions, so you should set RAPOOL to a low value in application- or file-owning regions (AORs or FORs).

Limitations

HPO requires CICS to be authorized, and some risks with MVS integrity are involved because a user-written module could be made to replace one of the CICS system initialization routines and run in authorized mode. This risk can be reduced by RACF protecting the CICS SDFHAUTH data set.

Use of HPO saves processor time, and does not increase real or virtual storage requirements or I/O contention. The only expense of HPO is the potential security exposure that arises because of a deficiency in validation.

Recommendations

The general recommendation is that all production systems with vetted applications can use HPO. It is totally application-transparent and introduces no function restrictions while providing a reduced pathlength through VTAM. In the case of VTAM, the reduced validation does not induce any integrity loss for the messages.

How implemented

The SVCs and use of HPO are specified in the system initialization table (SIT) and, if the default SVC numbers are acceptable, no tailoring of the system is required.

How monitored

There is no direct measurement of HPO. One way to tell if it is working is to take detailed measurements of processor usage with HPO turned on (SIT option) and with it turned off. Depending on the workload, you may not see much difference. Another way to check whether it is working is that you may see a small increase in the SRB scheduling time with HPO turned on.

RMF can give general information on processor usage. An SVC trace can show how HPO was used.

Note that you should be take care when using HPO in a system that is being used for early testing of a new application or CICS code (a new release or PUT). Much of the pathlength reduction is achieved by bypassing control block verification code in VTAM. Untested code might possibly corrupt the control blocks that CICS passes to VTAM, and unvalidated applications can lead to security exposure.

SNA transaction flows (MSGINTEG, and ONEWTE)

Within CICS, the MSGINTEG option can be used to control the communication requests and responses that are exchanged between the terminals in a network and the VTAM and NCP communications programs.

Effects

One of the options in Systems Network Architecture (SNA) is whether the messages exchanged between CICS and a terminal are to be in definite or exception response mode. Definite response mode requires both the terminal and CICS to provide acknowledgment of receipt of messages from each other on a one-to-one basis.

SNA also ensures message delivery through synchronous data link control (SDLC), so definite response is not normally required. Specifying message integrity (MSGINTEG) causes the sessions for which it is specified to operate in definite response mode.

In other cases, the session between CICS and a terminal operates in exception response mode, and this is the normal case.

In SNA, transactions are defined within brackets. A begin bracket (BB) command defines the start of a transaction, and an end bracket (EB) command defines the end of that transaction. Unless CICS knows ahead of time that a message is the last of a transaction, it must send an EB separate from the last message if a transaction terminates. The EB is an SNA command, and can be sent with the message, eliminating one required transmission to the terminal.

Specifying the ONEWTE option for a transaction implies that only one output message is to be sent to the terminal by that transaction, and allows CICS to send the EB along with that message. Only one output message is allowed if ONEWTE is specified and, if a second message is sent, the transaction is abended.

The second way to allow CICS to send the EB with a terminal message is to code the LAST option on the last terminal control or basic mapping support SEND command in a program. Multiple SEND commands can be used, but the LAST option must be coded for the final SEND in a program.

The third (and most common) way is to issue SEND without WAIT as the final terminal communication. The message is then sent as part of task termination.

You have the following options:

- Not specifying MSGINTEG
- Specifying MSGINTEG (which simply asks for definite response to be forced)

Where useful

The above options can be used in all CICS systems that use VTAM.

Limitations

The MSGINTEG option causes additional transmissions to the terminal. Transactions remain in CICS for a longer period, and tie up virtual storage and access to resources (primarily enqueues). MSGINTEG is required if the transaction must know that the message was delivered.

When MSGINTEG is specified, the TIOA remains in storage until the response is received from the terminal. This option can increase the virtual storage requirements for the CICS region because of the longer duration of the storage needs.

How implemented

With resource definition online (RDO) using the CEDA transaction, protection can be specified in the PROFILE definition by means of the MSGINTEG, and ONEWTE options. The MSGINTEG option is used with SNA LUs only. See the *CICS Resource Definition Guide* for more information about defining a PROFILE.

How monitored

You can monitor the use of the above options from a VTAM trace by examining the exchanges between terminals and CICS and, in particular, by examining the contents of the request/response header (RH).

SNA chaining (TYPETERM RECEIVESIZE, BUILDCHAIN, and SENDSIZE)

Systems Network Architecture (SNA) allows terminal messages to be chained, and lets large messages be split into smaller parts while still logically treating the multiple message as a single message.

Input chain size and characteristics are normally dictated by the hardware requirements of the terminal in question, and so the CEDA BUILDCHAIN and RECEIVESIZE attributes have default values which depend on device attributes. The size of an output chain is specified by the CEDA SENDSIZE attribute.

Effects

Because the network control program (NCP) also segments messages into 256-byte blocks for normal LU Type 0, 1, 2, and 3 devices, a SENDSIZE value of zero eliminates the overhead of output chaining. A value of 0 or 1536 is required for local devices of this type.

If you specify the CEDA SENDSIZE attribute for intersystem communication (ISC) sessions, this must match the CEDA RECEIVESIZE attribute in the other system. The CEDA SENDSIZE attribute or TCT BUFFER operand controls the size of the SNA element that is to be sent, and the CEDA RECEIVESIZEs need to match so that there is a corresponding buffer of the same size able to receive the element.

If you specify BUILDCHAIN(YES), CICS assembles a complete chain of elements before passing them to an application. If you do not specify BUILDCHAIN(YES), each individual RU is passed to an individual receive-any in the application. With SNA/3270 BMS does not work correctly if you do not specify BUILDCHAIN(YES).

If you are dealing with very large inbound elements that exceed a maximum of 32KB, you cannot use the BUILDCHAIN attribute or CHNASSY operand. You must use multiple individual RUs, and this extends the transaction life in the system.

Where useful

Chaining can be used in systems that use VTAM and SNA terminals of types that tolerate chaining.

Limitations

If you specify a low CEDA SENDSIZE value, this causes additional processing and real and virtual storage to be used to break the single logical message into multiple parts.

Chaining may be required for some terminal devices. Output chaining can cause flickering on display screens, which can annoy users. Chaining also causes additional I/O overhead between VTAM and the NCP by requiring additional VTAM subtasks and STARTIO operations. This additional overhead is eliminated

with applicable ACF/VTAM releases by making use of the large message performance enhancement option (LMPEO).

Recommendations

The CEDA RECEIVESIZE value for IBM 3274-connected display terminals should be 1024; for IBM 3276-connected display terminals it should be 2048. These values give the best line characteristics while keeping processor usage to a minimum.

How implemented

Chaining characteristics are specified in the CEDA DEFINE TYPETERM statement with the SENDSIZE, BUILDCHAIN, and RECEIVESIZE attributes.

How monitored

Use of chaining and chain size can be determined by examining a VTAM trace. You can also use the CICS internal and auxiliary trace facilities, in which the VIO ZCP trace shows the chain elements. Some of the network monitor tools such as NetView Performance Monitor (NPM) give this data.

Number of concurrent logon/logoff requests (OPNDLIM)

The OPNDLIM operand defines the number of concurrent VTAM logons and logoffs that are to be processed by CICS. In systems running ACF/VTAM Release 3.2 and later, this operand is not necessary and will be ignored. In all other instances this system initialization parameter limits the number of concurrent logon OPNDST and logoff CLSDST requests. The smaller this value, the smaller the amount of storage that is required during the open and close process.

Each concurrent logon/logoff requires storage in the CICS dynamic storage areas for the duration of that processing.

Effects

Particularly when logons are being done automatically with either the CICS CONNECT=AUTO facility or the VTAM LOGAPPL facility, large numbers of logons can occur at CICS startup or restart times. In systems running ACF/VTAM with a release prior to 3.2 this can require significant amounts of storage, which can be reduced with the OPNDLIM operand. In ACF/VTAM Release 3.2 and later systems, this operand is not necessary and will be ignored.

If an automatic logon facility is required, the LOGAPPL facility offers two advantages. It requires approximately 3500 bytes less storage in VTAM than the CONNECT=AUTO facility, and it logs terminals back on to CICS each time the device is activated to VTAM, rather than only at CICS initialization.

Where useful

The OPNDLIM system initialization parameter can be used in CICS systems that use VTAM as the terminal access method.

The OPNDLIM system initialization parameter can also be useful if there are times when all the user community tends to log on or log off at the same time, for example, during lunch breaks.

Limitations

If too low a value is specified for OPNDLIM, real and virtual storage requirements are reduced within CICS and VTAM buffer requirements may be cut back, but session initializations and terminations take longer.

Recommendations

Use the default value initially and make adjustments if statistics indicate that too much storage is required in your environment or that the startup time (DEFINE TYPETERM AUTOCONNECT attribute in CEDA) is excessive.

OPNDLIM should be set to a value not less than the number of LUs connected to any single VTAM line.

How implemented

OPNDLIM is a system initialization parameter.

How monitored

Logon and logoff activities are not reported directly by CICS or any measurement tools, but can be analyzed using the information given in a VTAM trace or VTAM display command.

Terminal scan delay (ICVTSD)

The terminal scan delay (ICVTSD) system initialization parameter determines the frequency with which CICS attempts to process terminal output requests.

In general, this value defines the time that the terminal control program must wait to process:

- Non-VTAM terminal I/O requests with WAIT specified
- Non-VTAM output deferred until task termination
- Automatic transaction initiation (ATI) requests
- VTAM terminal management, including output request handling, in busy CICS systems with significant application task activity.

This last case arises from the way that CICS scans active tasks.

On CICS non-VTAM systems, the delay value specifies how long the terminal control program must wait after an application terminal request, before it carries out a TCT scan. The value thus controls batching and delay in the associated processing of terminal control requests. In a low-activity system, it controls the dispatching of the terminal control program.

The batching of requests reduces processor time at the expense of longer response times. On CICS VTAM systems, it influences how quickly the terminal control program completes VTAM request processing, especially when the MVS high performance option (HPO) is being used.

Effects

VTAM

In VTAM networks, a low ICVTSD value does not cause full TCT scans because the input from or output to VTAM terminals is processed from the activate queue chain, and only those terminal entries are scanned.

With VTAM terminals, CICS uses bracket protocol to indicate that the terminal is currently connected to a transaction. The bracket is started when the transaction is initiated, and ended when the transaction is terminated. This means that there could be two outputs to the terminal per transaction: one for the data sent and one when the transaction terminates containing the end bracket. In fact, only one output is sent (except for WRITE/SEND with WAIT and definite response). CICS holds the output data until the next terminal control request or termination. In this way it saves processor cycles and line utilization by sending the message and end bracket or change direction (if the next request was a READ/RECEIVE) together in the same output message (PIU). When the system gets very busy, terminal control is dispatched less frequently and becomes more dependent upon the value specified in ICVTSD. Because CICS may not send the end bracket to VTAM for an extended period of time, the life of a transaction can be extended. This keeps storage allocated for that task for longer periods and potentially increases the amount of virtual storage required for the total CICS dynamic storage areas.

Setting ICVTSD to zero can overcome this effect.

Non-VTAM

ICVTSD is the major control on the frequency of full terminal control table (TCT) scanning of non-VTAM terminals. In active systems, a full scan is done approximately once every ICVTSD. The average extra delay before sending an output message should be about half this period.

In non-VTAM networks, partial scans occur for other reasons, such as an input arriving from a terminal, and any outputs for that line are processed at the same time. For that reason, a value of between 0.5 and one second is normally a reasonable setting for non-VTAM networks.

CICS scans application tasks first, unless there is an ICVTSD-driven scan. In a highly utilized system, input and output messages may be unreasonably delayed if too large a ICVTSD value is specified.

All networks

The ICVTSD parameter can be changed in the system initialization table (SIT) or through JCL parameter overrides. If you are having virtual storage constraint problems, it is highly recommended that you reduce the value specified in ICVTSD. A value of zero causes the terminal control task to be dispatched most frequently. If you also have a large number of non-VTAM terminals, this may increase the amount of nonproductive processor cycles. A value of 100—300 milliseconds may be more appropriate for that situation. In a pure VTAM environment, however, the overhead is not significant, unless the average transaction has a very short pathlength, and ICVTSD should be set to zero for a better response time and best virtual storage usage.

Where useful

The ICVTSD system initialization parameter can be used in all except very low-activity CICS systems.

Limitations

In TCAM systems, a low ICVTSD value can cause excessive processor time to be used in slower processor units, and can delay the dispatch of user tasks because too many full TCT scans have to be done. A high ICVTSD value can increase

response time by an average of one half of the ICVTSD value, and can tie up resources owned by the task because the task takes longer to terminate. This applies to conversational tasks.

In VTAM systems, a low value adds the overhead of scanning the activate queue TCTTE chain, which is normally a minor consideration. A high value in high-volume systems can increase task life and tie up resources owned by that task for a longer period of time; this can be a significant consideration.

A low, nonzero value of ICVTSD can cause CICS to be dispatched more frequently, which increases the overhead of performance monitoring.

Recommendations

Set ICVTSD to a value less than the region exit time interval (ICV), which is also in the system initialization table (see page 188). Use the value of zero in an environment that contains only VTAM terminals and consoles, unless your workload consists of many short transactions. ICVTSD=0 in a VTAM terminal-only environment is not recommended for a CICS workload consisting of low terminal activity but with high TASK activity. Periods of low terminal activity can lead to delays in CSTP being dispatched. Setting ICVTSD=100-500 resolves this by causing CSTP to be dispatched regularly. For non-VTAM systems, specify the value of zero only for small networks (1 through 30 terminals).

For almost all systems that are not "pure" VTAM, the range should be somewhere in the region of 100 milliseconds to 1000 milliseconds. ICVTSD can be varied between, say, 300 and 1000 milliseconds without a very significant effect on the response time, but increasing the value decreases the processor overhead. An ICVTSD larger than 1000 milliseconds may not give any further improvement in processor usage, at a cost of longer response times.

If ICVTSD is reduced, and, if there is ample processor resource, a small reduction in response time can be achieved. If you go below 250 milliseconds, any improvement in response time is likely to seem negligible to the end user and would have an increased effect on processor usage.

The recommended absolute minimum level, for systems that are not "pure" VTAM, is approximately 250 milliseconds or, in really high-performance, high-power systems that are "pure" VTAM, 100 milliseconds.

How implemented

The ICVTSD system initialization parameter is defined in units of milliseconds. Use the commands CEMT or EXEC CICS SET SYSTEM SCANDELAY (nnnn) to reset the value of ICVTSD.

In reasonably active systems, a nonzero ICVTSD virtually replaces ICV (see page 190) because the time to the next TCT full scan (non-VTAM) or sending of output requests (VTAM) is the principal influence on operating system wait duration.

How monitored

Use RMF to monitor task duration and processor requirements. The dispatcher domain statistics reports the value of ICVTSD.

Negative poll delay (NPDELAY)

NPDELAY in the TCT TYPE=LINE macro helps reduce unsolicited-input errors on TCAM lines.

NPDELAY and unsolicited-input messages in TCAM

Any CICS users who do not want unsolicited-input messages to be discarded should consider using the optional NPDELAY operand for the DFHTCT TYPE=LINE macro used to define each of the TCAM queues. This allows you to define a time interval during which CICS suspends the reading of messages from the respective TCAM queue following the receipt of unsolicited input. Upon completion of the preceding transaction associated with the same terminal as the unsolicited message, within the NPDELAY-defined interval, processing resumes normally.

Effects

If the preceding transaction fails to terminate during the NPDELAY interval, the X'87' unsolicited-input error condition is raised.

When NPDELAY is used, it is frequently advisable to define several input queues to CICS, each defined by a separate LINE entry but with each entry naming the same corresponding output queue using the OUTQ parameter. An equivalent number of queues must be defined to TCAM with the TPROCESS macro. This set of queues can then be processed as a "cascade" list within TCAM.

Where useful

When several queues are defined for TCAM-to-CICS processing, CICS can suspend the acceptance of input messages from one or more of the queues without completely stopping the flow of input from TCAM to CICS.

Choosing an appropriate value for NPDELAY is a matter of tuning. Even with the "cascade" list approach, some messages may be held up behind an unsolicited message. The objective should be to find the minimum value that can be specified for NPDELAY which is sufficient to eliminate the unsolicited-input errors.

Compression of output terminal data streams

For output messages, CICS provides user exits with access to the entire output data stream. User code can be written to remove redundant characters from the data stream before the data stream is sent to the terminal. This technique can produce a dramatic improvement in response times if the proportion of characters not needed is large, because telecommunication links are usually the slowest paths in the network.

Limitations

Some additional processor cycles are required to process the exit code, and the coding of the exit logic also requires some effort. Use of a compression exit reduces the storage requirements of VTAM or TCAM and NCP, and reduces line transmission time.

Recommendations

The simplest operation is to replace redundant characters, especially blanks, with a repeat-to-address sequence in the data stream for 3270-type devices.

Note: The repeat-to-address sequence is not handled very quickly on some types of 3270 cluster controller. In some cases, alternatives may give superior performance. For example, instead of sending a repeat-to-address sequence for a series of blanks, you should consider sending an ERASE and then set-buffer-address sequences to skip over the blank areas. This is satisfactory if nulls are acceptable in the buffer as an alternative to blanks.

Another technique for reducing the amount of data transmitted is to turn off any modified data tags on protected fields in an output data stream. This eliminates the need for those characters to be transmitted back to the processor on the next input message, but you should review application dependencies on those fields before you try this.

There may be other opportunities for data compression in individual systems, but you may need to investigate the design of those systems thoroughly before you can implement them.

How implemented

The global user exits used to compress terminal messages are the XZCOUT1 exit for VTAM devices, and the XTCTOUT exit for TCAM-supported devices. See the *CICS Customization Guide* for programming information.

How monitored

The contents of output terminal data streams can be examined in either a VTAM or TCAM trace.

Automatic installation of terminals

During autoinstall processing, CICS obtains storage from the control subpool in the extended CICS dynamic storage area (ECDSA), to handle each autoinstall request. The amount of virtual storage obtained is mainly determined by the length of the CINIT request unit, which varies for different LU types. For a typical autoinstall request from an LU6.2 terminal, the amount of dynamic virtual storage obtained is between 120 to 250 bytes.

Overall, the principal consumer of CICS resource in autoinstall processing is the autoinstall task (CATA) itself. If, for some reason, the autoinstall process is not proceeding at the rate expected during normal operations, there is a risk that the system could be filled with CATA transaction storage.

Maximum concurrent autoinstalls (AIQMAX)

This system initialization parameter codes the maximum number of devices that can be queued concurrently for autoinstall.

The AIQMAX value does not limit the total number of devices that can be autoinstalled.

The restart delay parameter (AIRDELAY)

This system initialization parameter specifies whether you want autoinstalled terminal definitions to be retained by CICS across a restart. The value of the restart delay is specified as “hhmmss” and the default is “000700”, which is seven minutes. This means that if a terminal does not log on to CICS within seven minutes after an emergency restart, its terminal entry is scheduled for deletion.

Setting the restart delay to zero means that you do not want CICS to re-install the autoinstalled terminal entries from the global catalog during emergency restart. In this case, CICS does not write the terminal entries to the catalog while the terminal is being autoinstalled. This can have positive performance effects on the following processes:

Autoinstall By eliminating the I/O activity, autoinstall has a shorter pathlength and becomes more processor-intensive. So, in general, the time taken to autoinstall a terminal is reduced. However, the response time of other tasks may increase slightly because CATA has a high priority and does not have to wait for as much I/O activity.

Emergency and warm restart When no autoinstalled terminal entries are cataloged, CICS has to restore fewer entries from the GCD during emergency restart. Thus, if you have a large number of autoinstalled terminals, the restart time can be significantly improved when restart delay is set to zero.

Normal shutdown CICS deletes AI terminal entries from the GCD during normal shutdown unless they were not cataloged (AIRDELAY=0) and the terminal has not been deleted. If the restart delay is set to zero, CICS has not cataloged terminal entries when they were autoinstalled, so they are not deleted. This can reduce normal shutdown time.

XRF takeover The system initialization parameter, AIRDELAY, should not affect XRF takeover. The tracking process still functions as before regardless of the value of the restart delay. Thus, after a takeover, the alternate system still has all the autoinstalled terminal entries. However, if a takeover occurs before the catchup process completes, some of the autoinstalled terminals have to log on to CICS again. The alternate CICS system has to rely on the catalog to complete the catchup process and, if the restart delay is set to zero in the active system, the alternate system is not able to restore the autoinstalled terminal entries that have not been tracked. Those terminals have to log on to the new CICS system, rather than being switched or rebound after takeover.

You have to weigh the risk of having some terminal users log on again because tracking has not completed, against the benefits introduced by setting the restart delay to zero. Because catchup takes only a few minutes, the chance of such a takeover occurring is usually small.

The delete delay parameter (AILDELAY)

The delete delay system initialization parameter lets you control how long an autoinstalled terminal entry remains available after the terminal has logged off. The default value of zero means that the terminal entry is scheduled for deletion as soon as the terminal is logged off. Otherwise, CICS schedules the deletion of the TCTTE as a timer task.

In general, setting the delete delay to a nonzero value can improve the performance of CICS when many autoinstalled terminals are logging on and off during the day. However, this does mean that unused autoinstalled terminal entry storage is not freed for use by other tasks until the delete delay interval has expired. This parameter provides an effective way of defining a terminal whose storage lifetime is somewhere between that of an autoinstalled terminal and a statically defined terminal.

The effect of setting the delete delay to a nonzero value can have different effects depending on the value of the restart delay:

Nonzero restart delay When the restart delay is nonzero, CICS catalogs autoinstalled terminal entries in the global catalog.

If the delete delay is nonzero as well, CICS retains the terminal entry so that it is re-used when the terminal logs back on. This can eliminate the overhead of:

- Deleting the terminal entry in virtual storage
- An I/O to the catalog and recovery log
- Re-building the terminal entry when the terminal logs on again.

Zero restart delay When the restart delay is zero, CICS does not catalog autoinstalled terminal entries in the global catalog whatever value is specified for the delete delay.

If the delete delay is nonzero, CICS retains the terminal entry so that it is re-used when the terminal logs back on. This can save the overhead of deleting the terminal entry in virtual storage and the rebuilding of the terminal entry when the terminal logs on again.

Effects

You can control the use of resource by autoinstall processing in three ways:

1. By using the transaction class limit to restrict the number of autoinstall tasks that can concurrently exist (see page 286).
2. By using the CATA and CATD transactions to install and delete autoinstall terminals dynamically. If you have a large number of devices autoinstalled, shutdown can fail due to the MXT system initialization parameter being reached or CICS becoming short on storage. To prevent this possible cause of shutdown failure, you should consider putting the CATD transaction in a class of its own to limit the number of concurrent CATD transactions.
3. By specifying AIQMAX to limit the number of devices that can be queued for autoinstall. This protects against abnormal consumption of virtual storage by the autoinstall process, caused as a result of some other abnormal event.

   If this limit is reached, the AIQMAX system initialization parameter affects the LOGON and BIND processing by CICS. CICS requests VTAM to stop passing LOGON and BIND requests to CICS. VTAM holds such requests until CICS indicates that it can accept further LOGONs and BINDs (this occurs when CICS has processed a queued autoinstall request).

Recommendations

If the autoinstall process is noticeably slowed down by the AIQMAX limit, raise it. If the CICS system shows signs of running out of storage, reduce the AIQMAX limit. If possible, set the AIQMAX system initialization parameter to a value higher than that reached during normal operations.

In a *non-XRF* environment, settings of (restart delay=0) and (delete delay= hhmmss>0) are the most efficient for processor and DASD utilization. However, this efficiency is gained at a cost of virtual storage, because the TCT entries are not deleted until the delay period expires.

A value of zero for both restart delay and delete delay is the best overall setting for many systems from an overall performance and virtual-storage usage point of view.

If restart delay is greater than zero (cataloging active), the performance of autoinstall is significantly affected by the definition of the global catalog (DFHGCD) . The default buffer specifications used by VSAM may not be sufficient in a high activity system.

Because a considerable number of messages are sent to transient data during logon and logoff, the performance of these output destinations should also be taken into consideration.

In an *XRF* environment, a restart delay value of greater than zero should give better performance when catchup of a large number of autoinstalled terminals is necessary.

How monitored

Monitor the autoinstall rate during normal operations by inspecting the autoinstall statistics regularly.

Chapter 17. LU6.2 sessions limit

Within the LU6.2 architecture CICS supported a LU6.2 session limit for each region of 46656. Functional enhancements have allowed this limit to be extended to 93312 sessions.

No immediate change to performance occurs when the original limit is passed, but the following considerations should be made when having large numbers of sessions connected to a single CICS region:

- CICS startup time will increase if more sessions are installed at initial startup time
- CICS restart time will increase with the additional number of sessions
- You should ensure that the global catalogue is large enough to hold the additional number of records
- You should ensure that the log structures can accommodate the extra load imposed by so many sessions.
- You should distribute the high number of sessions across multiple CICS regions, if possible. From a resource contention point of view this can relieve CPU TCB contention and I/O contention to the catalogue.
- It is also advisable to spread high numbers of sessions across different CICS regions for availability reasons, so that the whole network is not lost if failure occurs.

When a higher number of sessions are installed into a region you can potentially experience higher transaction rates and it is recommended that you undertake a resource capacity planning exercise. A storage capacity planning exercise should be carried out, anyway for the increase in storage requirement for any additional sessions.

Chapter 18. CICS Web support

This chapter includes the following topics:

CICS Web performance in a sysplex

The dynamic routing facility is extended to provide mechanisms for dynamically routing program—link requests received from outside CICS. The target program of a CICS Web application can be run anywhere in a sysplex by dynamically routing the EXEC CICS LINK to the target application. Web bridge transactions should either be not routed or always routed to the same region so that there are major affinitites. Using CICSPlex SM to route the program-link requests, the transaction ID becomes much more significant because CICSPlex SM's routing logic is transaction-based. CICSPlex SM routes each DPL request according to the rules specified for its associated transaction. This dynamic routing means that there is extra pathlength for both routed and nonrouted links, and routing links.

CICSPlex SM allows you to take advantage of two dynamic routing models:

The hub model
: A *hierarchical* system used traditionally with CICS dynamic transaction routing. Routing is controlled by one region,

The distributed model
: Each region may be both a routing region and a target region. A routing region runs in each region.

In addition, you can define your own algorithm.

Analyzer and converter programs must run in the same region as the instance of DFHWBBLI which invokes them, which in the case of CICS Web support, is the CICS region on which the HTTP request is received.

If the Web API is being used by the application program to process the HTTP request and build the HTTP response, the application program must also run in the same CICS region as the instance of DFHWBBLI which is linking to it.

In a typical current scenario, a Web-based business transaction might be implemented as a pseudoconversational CICS application. The initial request from the browser invokes a CICS transaction that does some setup work, returns a page of HTML to the browser, and ends. Subsequent requests are handled by other CICS transactions (or by invoking further the same transaction). The CICS application is responsible for maintaining state data between requests.

Using Business Transaction Services, a Web-based business transaction could be implemented as a BTS process. An advantage of this is that state data is maintained by BTS.

CICS Web support performance in a single address space

The additional cost of using the WEB API commands, compared with the cost of commarea manipulation as a means of processing the received requests and building the HTTP responses, can be in the range of from 6% to 12%. Using very large number of bookmarks in the building of CICS documents can add more to this figure. However, the ease of programming offered by the WEB API commands makes the cost worthwhile.;

The use of HTTP persistent connection operation, which is supported by most client Web browsers, is also supported by the CICS Web interface in CICS Transaction Server for OS/390 Release 3, and very significant savings in CICS and TCP/IP CPU cost, plus response times at the browser, are typically made by activating this feature in the TCPIPSERVICE definition. For more information about using the TCPIPSERVICE definition, see the *CICS Internet Guide*

CICS Web use of DOCTEMPLATE resources

In releases of CICS prior to CICS Transaction Server for OS/390 Release 3, CICS web applications use CICS HTML templates to facilitate the building of HTTP responses. These HTML templates had to reside in one MVS partitioned data set, designated by the DFHHTML DD statement n the CICS startup job. In CICS Transaction Server for OS/390 Release 3, CICS HTML template support has been extended, and each template should now be defined in the CICS CSD as a DOCTEMPLATE. When defining the DOCTEMPLATE, systems administrators can store their HTML templates in:

- Extrapartition transient data
- Temporary Storage
- CICS loaded programs
- MVS partitioned data sets or PDSEs
- Another location, invoking a user-written program to load the template from that location (for example, DB2 or another database manager).

To achieve optimum performance when using templates, you should ensure you have defined the template as DOCTEMPLATE and installed the definition before using it, especially when using the DFHWBTL program. If the template is not preinstalled when this program is used, DFHWBTL attempts to install it for you, assuming that it is a member of the partitioned dataset referenced by the DFHHTML DD statement.

The fastest results can be achieved by storing your templates as CICS load modules. For more information about this, see the *CICS Internet Guide*. These modules are managed like other CICS loaded programs and may be flushed out by program compression when storage is constrained.

CICS Web support use of temporary storage

CICS Web support now uses CICS temporary storage to store the inbound HTTP request and any outbound response built using the new Web API. You should define the characteristics of the TS queue used by CICS Web support for each TCPIPSERVICE by defining a TS model for the TS Q prefix identified on the relevant TCPIPSERVICE definition. A sample TS model named DFHWEB is provided in group DFHWEB, which defines the characteristics of a TS Queue with the prefix DFHWEB. The default definition uses MAIN temporary storage to mimimize the amount of I/O needed to process CICS Web requests. For those HTTP requests and responses which handle small amounts of data, this may be acceptable. If CICS Web support is being used to transfer large amounts of data, MAIN TS may not be appropriate, so the relevant TCPIPSERVICE should specify a TS Q prefix matching a model which uses AUXILIARY temporary storage.

When the CICS Web Business Logic Interface is used, the TS queue prefix is always DFHWEB.

CICS Web support of HTTP 1.0 persistent connections

In most circumstances CICS Web performance will be improved by enabling support of the HTTP 1.0 Keepalive header.

To enable CICS support of this header, you have to specify NO or a numeric value for the SOCKET CLOSE keyword on the relevant TCPIPSERVICE definition; if NO or a numeric value is specified, and the incoming HTTP request contains the Keepalive header, CICS keeps the socket open in order to allow further HTTP requests to be sent by the Web Browser. If a numeric value is specified, the interval between receipt of the last HTTP request and arrival of the next must be less than the interval specified on the TCPIPSERVICE, else CICS closes the socket. Some HTTP proxy servers do not allow the HTTP 1.0 Keepalive header to be passed to the end server (in this case, CICS), so Web Browsers which wish to use this header may not be able to pass it to CICS if the HTTP request arrives via such an HTTP proxy server.

CICS Web security

If Secure Sockets Layer is used to make CICS Web transactions more secure, there will be a significant increase in pathlength for these transactions. This increase can be minimized by use of the HTTP 1.0 Keepalive header. Keeping the socket open removes the need to perform a full SSL handshake on the second and any subsequent HTTP request. If CICS or the Web Browser closes the socket, the SSL handshake has to be executed again.

CICS Web 3270 support

Use of the HTTP 1.0 Keepalive header can improve the performance of CICS Web 3270 support, by removing the need for the Web Browser to open a new sockets connection for each leg of the 3270 conversation or pseudoconversation.

Secure sockets layer support

Transactions using Secure Sockets Layer for Web security will see an increase in pathlength because of the SSL handshake that occurs when the socket connection is established. Encryption and decryption impact performance, but degradation can be minimized by:

- Installing the appropriate cryptographic hardware.
- Making use of the HTTP 1.0 keepalive header.
- Making the CICS region as large as possible (The SSL support can use large amounts of non-CICS storage.)
- Only using SSL for applications that really need to use encrypted data flows.

You should also only use client authentication (SSL(CLIENTAUTH) in the TCPIPSERVICE definition) when you really need your clients to identify themselves with a client certificate. This is because client authentication involves more network interchanges during the SSL handshake, and more internal CICS processing to handle the received certificate. This includes a search of the external security manager's database to locate a user ID to associate with the certificate.

Chapter 19. VSAM and file control

This chapter discusses performance tuning issues related to VSAM and file control.

VSAM considerations: general objectives

Tuning consists of providing a satisfactory level of service from a system at an acceptable cost. A satisfactory service, in the case of VSAM, is likely to be obtained by providing adequate buffers to minimize physical I/O and, at the same time, allowing several operations concurrently on the data sets.

The costs of assigning additional buffers and providing for concurrent operations on data sets are the additional virtual and real storage that is required for the buffers and control blocks.

Several factors influence the performance of VSAM data sets. The rest of this section reviews these and the following sections summarize the various related parameters of file control.

Note that, in this section, a distinction is made between "files" and "data sets":

- A "file" means a view of a data set as defined by an installed CICS file resource definition and a VSAM ACB.
- A "data set" means a VSAM "sphere", including the base cluster with any associated AIX® paths.

Local shared resources (LSR) or Nonshared resources (NSR)

The first decision to make for each file is whether to use LSR or NSR for its VSAM buffers and strings. It is possible to use up to eight separate LSR pools for file control files. There is also a decision to make on how to distribute the data sets across the LSR pools.

Note that all files opened for access to a particular VSAM data set normally must use the same resource type: see "Data set name sharing" on page 228.

CICS provides separate LSR buffer pools for data and index records. If only data buffers are specified, only one set of buffers are built and used for both data and index records.

LSR files share a common pool of buffers and a common pool of strings (that is, control blocks supporting the I/O operations). Other control blocks define the file and are unique to each file or data set. NSR files or data sets have their own set of buffers and control blocks.

Some important differences exist between NSR and LSR in the way that VSAM allocates and shares the buffers.

In NSR, the minimum number of data buffers is STRNO + 1, and the minimum index buffers (for KSDSs and AIX paths) is STRNO. One data and one index buffer are preallocated to each string, and one data buffer is kept in reserve for CI splits. If there are extra data buffers, these are assigned to the first sequential operation; they may also be used to speed VSAM CA splits by permitting chained I/O operations. If there are extra index buffers, they are shared between the strings and are used to hold high-level index records, thus providing an opportunity for saving physical I/O.

In LSR, there is no preallocation of buffers to strings, or to particular files or data sets. When VSAM needs to reuse a buffer, it picks the buffer that has been referenced least recently. Strings are always shared across all data sets.

Before issuing a read to disk when using LSR, VSAM first scans the buffers to check if the control interval it requires is already in storage. If so, it may not have to issue the read. This buffer "lookaside" can reduce I/O significantly.

Another important difference between LSR and NSR is in concurrent access to VSAM CIs. NSR allows multiple copies of a CI in storage; you can have one (but only one) string updating a CI and other strings reading different copies of the same CI. In LSR, there is only one copy of a CI in storage; the second of the requests must queue until the first operation completes. LSR permits several read operations to share access to the same buffer, but updates require exclusive use of the buffer and must queue until a previous update or previous reads have completed; reads must wait for any update to finish. It is possible, therefore, that transactions with concurrent browse and update operations that run successfully with NSR may, with LSR, hit a deadlock as the second operation waits unsuccessfully for the first to complete.

Transactions should always be designed and programmed to avoid deadlocks. For further discussions, see the *CICS Application Programming Guide*.

LSR has significant advantages, by providing:

- More efficient use of virtual storage because buffers and strings are shared.
- Better performance because of better buffer lookaside, which can reduce I/O operations.
- Self-tuning because more buffers are allocated to busy files and frequently referenced index control intervals are kept in its buffers.
- Better read integrity because there is only one copy of a CI in storage.

- Use of synchronous file requests and a UPAD exit. CA and CI splits for LSR files do not cause either the subtask or main task to wait. VSAM takes the UPAD exit while waiting for physical I/O, and processing continues for other CICS work during the CA/CI split.

  File control requests for NSR files are done asynchronously, however, and still cause the CICS main task or subtask to stop during a split.

NSR, on the other hand:

- Allows for specific tuning in favor of a particular data set
- Can provide better performance for sequential operations.

The general recommendation is to use LSR for all VSAM data sets except where you have one of the following situations:

- A file is very active but there is no opportunity for lookaside because, for instance, the file is very large.
- High performance is required by the allocation of extra index buffers.
- Fast sequential browse or mass insert is required by the allocation of extra data buffers.
- Control area (CA) splits are expected for a file, and extra data buffers are to be allocated to speed up the CA splits.

If you have only one LSR pool, a particular data set cannot be isolated from others using the same pool when it is competing for strings, and it can only be isolated when it is competing for buffers by specifying unique CI sizes. In general, you get more self-tuning effects by running with one large pool, but it is possible to isolate busy files from the remainder or give additional buffers to a group of high performance files by using several pools. It is possible that a highly active file has more successful buffer lookaside and less I/O if it is set up as the only file in an LSR subpool rather than using NSR. Also the use of multiple pools eases the restriction of 255 strings for each pool.

Number of strings

The next decision to be made is the number of concurrent accesses to be supported for each file and for each LSR pool.

This is achieved by specifying VSAM “strings”. A string is a request to a VSAM data set requiring “positioning” within the data set. Each string specified results in a number of VSAM control blocks (including a “placeholder”) being built.

VSAM requires one or more strings for each concurrent file operation. For nonupdate requests (for example, a READ or BROWSE), an access using a base needs one string, and an access using an AIX needs two strings (one to hold position on the AIX and one to hold position on the base data set). For update requests where no upgrade set is involved, a base still needs one string, and a path two strings. For update requests where an upgrade set is involved, a base needs 1+*n* strings and a path needs 2+*n* strings, where *n* is the number of members in the upgrade set (VSAM needs one string per upgrade set member to hold position). Note that, for each concurrent request, VSAM can reuse the *n* strings required for upgrade set processing because the upgrade set is updated serially. See “CICS calculation of LSR pool parameters” on page 227.

A simple operation such as read direct frees the string or strings immediately, but a read for update, mass insert, or browse retains them until a corresponding update, unlock, or end browse is performed.

The interpretation of the STRNO parameter by CICS and by VSAM differs depending upon the context:

- The equivalent STRINGS parameter of the file definition has the same meaning as the STRNO in the VSAM ACB for NSR files: that is, the actual number of concurrent outstanding VSAM requests that can be handled. When AIX paths or upgrade sets are used, the actual number of strings which VSAM allocates to support this may be greater than the STRINGS value specified.
- The equivalent STRINGS parameter of the LSR pool definition (LSRPOOL) has the same meaning as the STRNO in the VSAM BLDVRP macro: that is, the absolute number of strings to be allocated to the resource pool. Unless an LSR pool contains only base data sets, the number of concurrent requests that can be handled is less than the STRINGS value specified.

Note: There are some special considerations for setting the STRINGS value for an ESDS file (see "Number of strings considerations for ESDS files" on page 225).

For LSR, it is possible to specify the precise numbers of strings, or to have CICS calculate the numbers. The number specified in the LSR pool definition is the actual number of strings in the pool. If CICS is left to calculate the number of strings, it derives the pool STRINGS from the RDO file definition and interprets this, as with NSR, as the actual number of concurrent requests. (For an explanation of CICS calculation of LSR pool parameters, see "CICS calculation of LSR pool parameters" on page 227.)

You must decide how many concurrent read, browse, updates, mass inserts, and so on you need to support.

If access to a file is read only with no browsing, there is no need to have a large number of strings; just one may be sufficient. Note that, while a read operation only holds the VSAM string for the duration of the request, it may have to wait for the completion of an update operation on the same CI.

In general (but see"Number of strings considerations for ESDS files" on page 225) where some browsing or updates are used, STRINGS should be set to 2 or 3 initially and CICS file statistics should be checked regularly to see the proportion of wait-on-strings encountered. Wait-on-strings of up to 5% of file accesses would usually be considered quite acceptable. You should not try, with NSR files, to keep wait-on-strings permanently zero.

CICS manages string usage for both files and LSR pools. For each file, whether it uses LSR or NSR, CICS limits the number of concurrent VSAM requests to the STRINGS= specified in the file definition. For each LSR pool, CICS also prevents more requests being concurrently made to VSAM than can be handled by the strings in the pool. Note that, if additional strings are required for upgrade-set processing at update time, CICS anticipates this requirement by reserving the additional strings at read-for-update time. If there are not enough file or LSR pool strings available, the requesting task waits until they are freed. The CICS statistics give details of the string waits.

When deciding the number of strings for a particular file, consider the maximum number of concurrent tasks. Because CICS command level does not allow more than one request to be outstanding against a particular data set from a particular task, there is no point in allowing strings for more concurrent requests.

If you want to distribute your strings across tasks of different types, the transaction classes may also be useful. You can use transaction class limits to control the transactions issuing the separate types of VSAM request, and for limiting the number of task types that can use VSAM strings, thereby leaving a subset of strings available for other uses.

All placeholder control blocks must contain a field long enough for the largest key associated with any of the data sets sharing the pool. Assigning one inactive file that has a very large key (primary or alternate) into an LSR pool with many strings may use excessive storage.

Number of strings considerations for ESDS files

There are some special performance considerations when choosing a STRINGS value for an ESDS file.

If an ESDS is used as an 'add-only' file (that is, it is used only in write mode to add records to the end of the file), a string number of 1 is strongly recommended. Any string number greater than 1 can significantly affect performance, because of exclusive control conflicts that occur when more than one task attempts to write to the ESDS at the same time.

If an ESDS is used for both writing and reading, with writing, say, being 80% of the activity, it is better to define two file definitions—using one file for writing and the other for reading.

Size of control intervals

The size of the data set control intervals is not an parameter specified to CICS; it is defined through VSAM AMS. However, it can have a significant performance effect on a CICS system that provides access to the control interval.

In general, direct I/O runs slightly more quickly when data CIs are small, whereas sequential I/O is quicker when data CIs are large. However, with NSR files, it is possible to get a good compromise by using small data CIs but also assigning extra buffers, which leads to chained and overlapped sequential I/O. However, all the extra data buffers get assigned to the first string doing sequential I/O.

VSAM functions most efficiently when its control areas are the maximum size, and it is generally best to have data CIs larger than index CIs. Thus, typical CI sizes for data are 4KB to 12KB and, for index, 1KB to 2KB.

In general, you should specify the size of the data CI for a file, but allow VSAM to select the appropriate index CI to match. An exception to this is if key compression turns out to be less efficient than VSAM expects it to be. In this case, VSAM may select too small an index CI size. You may find an unusually high rate of CA splits occurring with poor use of DASD space. If this is suspected, specify a larger index CI.

In the case of LSR, there may be a benefit in standardizing on the CI sizes, because this allows more sharing of buffers between files and thereby allow a lower total number of buffers. Conversely, there may be a benefit in giving a file unique CI sizes to prevent it from competing for buffers with other files using the same pool.

Try to keep CI sizes at 512, 1KB, 2KB, or any multiple of 4KB. Unusual CI sizes like 26KB or 30KB should be avoided. A CI size of 26KB does not mean that physical block size will be 26KB; the physical block size will most likely be 2KB in this case (it is device-dependent).

Number of buffers (NSR)

The next decision is the number of buffers to be provided for each file. Enough buffers must be provided to support the concurrent accesses specified in the STRINGS parameter for the file (in fact VSAM enforces this for NSR).

Specify the number of data and index buffers for NSR using the DATABUFFER and INDEXBUFFER parameters of the file definition. It is important to specify sufficient index buffers. If a KSDS consists of just one control area (and, therefore, just one index CI), the minimum index buffers equal to STRINGS is sufficient. But when a KSDS is larger than this, at least one extra index buffer needs to be specified so that at least the top level index buffer is shared by all strings. Further index buffers reduces index I/O to some extent.

DATABUFFERS should generally be the minimum at STRINGS + 1, unless the aim is to enable overlapped and chained I/O in sequential operations or it is necessary to provide the extra buffers to speed up CA splits.

Note that when the file is an AIX path to a base, the same INDEXBUFFERS (if the base is a KSDS) and DATABUFFERS are used for AIX and base buffers (but see "Data set name sharing" on page 228).

Number of buffers (LSR)

The set of buffers of one size in an LSR pool is called a "subpool." The number of buffers for each subpool is controlled by the DATA and INDEX parameters of the LSRPOOL definition It is possible to specify precise numbers or to have CICS calculate the numbers. (The method used by CICS to calculate the number of buffers is described under "CICS calculation of LSR pool parameters" on page 227.)

Allowing CICS to calculate the LSR parameters is easy but it incurs additional overhead (when the first file that needs the LSR pool is opened) to build the pool. Consider the following factors if you allow CICS to calculate an LSR pool:

- CICS must read the VSAM catalog for every file that is specified to use the pool. This in itself may be an unacceptable overhead.
- The overhead is significantly increased if the data sets involved are migrated at the time that CICS performs the calculation. This is because, to enable CICS to read the VSAM catalog for each data set associated with the LSR pool, each data set has to be recalled.

  Not only can a single recall cause a significant delay for the task that caused the recall, but it is a synchronous operation that delays other activities that CICS is running under the same TCB.

  You can avoid these delays by designing your SMS storage classes and migration policies to avoid CICS data sets being migrated..

  CICS outputs an information message, DHFC0989, when a recall is necessary, effectively advising you that the consequent delay is not an error situation.
- An LSR pool calculated by CICS cannot be fine-tuned by specifying actual sizes for each buffer.

When making changes to the size of an LSR pool, refer to the CICS statistics before and after the change is made. These statistics show whether the proportion of VSAM reads satisfied by buffer lookaside is significantly changed or not.

In general, you would expect to benefit more by having extra index buffers for lookaside, and less by having extra data buffers. This is a further reason for standardizing on LSR data and index CI sizes, so that one subpool does not have a mix of index and data CIs in it.

Note: Data and index buffers are specified separately with the LSRPOOL definition. Thus, there is not a requirement to use CI size to differentiate between data and index values.

Take care to include buffers of the right size. If no buffers of the required size are present, VSAM uses the next larger buffer size.

CICS calculation of LSR pool parameters

If you have not specified LSR parameters for a pool, CICS calculates for you the buffers and strings required. To do this, it scans all the installed file resource definitions for files specified to use the pool. For each, it uses:

- From the CICS file resource definitions:
  - The number of strings, as specified on the STRINGS parameter
- From the VSAM catalog:
  - The levels of index for each of these files
  - The CI sizes
  - The keylengths for the base, the path (if it is accessed through an AIX path), and upgrade set AIXs.

Note: If you have specified only buffers or only strings, CICS performs the calculation for what you have not specified.

The following information helps you calculate the buffers required. A particular file may require more than one buffer size. For each file, CICS determines the buffer sizes required for:

- The data component
- The index component (if a KSDS)
- The data and index components for the AIX (if it is an AIX path)
- The data and index components for each AIX in the upgrade set (if any).

The number of buffers for each is calculated as follows:

- For data components (base and AIX) = (STRINGS= in the file resource definition entry) + 1
- For index components (base and AIX) = (STRINGS= in the file resource definition entry) + (the number of levels in the index) – 1
- For data and index components for each AIX in the upgrade set, one buffer each.

When this has been done for all the files that use the pool, the total number of buffers for each size is:

- Reduced to either 50% or the percentage specified in the SHARELIMIT in the LSRPOOL definition. The SHARELIMIT parameter takes precedence.
- If necessary, increased to a minimum of three buffers.
- Rounded up to the nearest 4KB boundary.

To calculate the number of strings, CICS determines the number of strings to handle concurrent requests for each file as the sum of:

- STRINGS parameter value for the base
- STRINGS parameter value for the AIX (if it is an AIX path)
- *n* strings if there is an upgrade set (where *n* is the number of members in the upgrade set).

When the strings have been accumulated for all files, the total is:

- Reduced to either 50% or the percentage specified in the SHARELIMIT parameter in the LSR pool definition. The SHARELIMIT parameter takes precedence.
- Reduced to 255 (the maximum number of strings allowed for a pool by VSAM).
- Increased to the largest specified STRINGS value for a particular file.

The parameters calculated by CICS are shown in the CICS statistics.

Switching data sets from RLS mode to LSR mode

Although it is not generally recommended, there may be occasions when you need to switch a data set from RLS mode to non-RLS mode (for example, to read-only LSR mode during a batch update). This could lead to the LSR pools that are not explicitly defined, and which CICS builds using default values, not having sufficient resources to support files switched to LSR mode after the pool has been built.

To avoid files failing to open because of the lack of adequate resources, you can specify that CICS should include files opened in RLS mode when it is calculating the size of an LSR pool using default values. To specify the inclusion of files defined with RLSACCESS(YES) in an LSR pool being built using values that CICS calculates, use the RLSTOLSR=YES system initialization parameter (RLSTOLSR=NO is the default)

See the *CICS System Definition Guide* for more information about the RLSTOLSR parameter.

Data set name sharing

Data set name (DSN) sharing (MACRF=DSN specified in the VSAM ACB) is the default for all VSAM data sets. It causes VSAM to create a single control block structure for the strings and buffers required by all the files that relate to the same base data set cluster, whether as a path or direct to the base. VSAM makes the connection at open time of the second and subsequent files. Only if DSN sharing is specified, does VSAM realize that it is processing the same data set.

This single structure:

- Provides VSAM update integrity for multiple ACBs updating one VSAM data set
- Allows the use of VSAM share options 1 or 2, while still permitting multiple update ACBs within the CICS region
- Saves virtual storage.

DSN sharing is the default for files using both NSR and LSR. The only exception to this default is made when opening a file that has been specified as read-only (READ=YES or BROWSE=YES) and with DSNSHARING(MODIFYREQS) in the file resource definition. CICS provides this option so that a file (represented by an

installed file resource definition) can be isolated from other users of that same data set in a different LSR pool or in NSR by suppressing DSN sharing. CICS ignores this parameter for files with update, add, or delete options because VSAM would not then be able to provide update integrity if two file control file entries were updating the same data set concurrently.

The NSRGROUP= parameter is associated with DSN sharing. It is used to group together file resource definitions that are to refer to the same VSAM base data set. NSRGROUP=name has no effect for data sets that use LSR.

When the first member of a group of DSN-sharing NSR files is opened, CICS must specify to VSAM the total number of strings to be allocated for all file entries in the group, by means of the BSTRNO value in the ACB. VSAM builds its control block structure at this time regardless of whether the first data set to be opened is a path or a base. CICS calculates the value of BSTRNO used at the time of the open by adding the STRINGS values in all the files that share the same NSRGROUP= parameter.

If you do not provide the NSRGROUP= parameter, the VSAM control block structure may be built with insufficient strings for later processing. This should be avoided for performance reasons. In such a case, VSAM invokes the dynamic string addition feature to provide the extra control blocks for the strings as they are required, and the extra storage is not released until the end of the CICS run.

AIX considerations

For each AIX defined with the UPGRADE attribute, VSAM upgrades the AIX automatically when the base cluster is updated.

For NSR, VSAM uses a special set of buffers associated with the base cluster to do this. This set consists of two data buffers and one index buffer, which are used serially for each AIX associated with a base cluster. It is not possible to tune this part of the VSAM operation.

For LSR, VSAM uses buffers from the appropriate subpool.

Care should be taken when specifying to VSAM that an AIX should be in the upgrade set. Whenever a new record is added, an existing record deleted, or a record updated with a changed attribute key, VSAM updates the AIXs in the upgrade set. This involves extra processing and extra I/O operations.

Situations that cause extra physical I/O

Listed below are some situations that can lead to a lot of physical I/O operations, thus affecting both response times and associated processor pathlengths:

- When a KSDS is defined with SHROPT of 4, all direct reads cause a refresh of both index and data buffers (to ensure latest copy).
- Any sequence leading to CICS issuing ENDREQ invalidates all data buffers associated with the operation. This may occur when you end a get-update (without the following update), a browse (even a start browse with a no-record-found response), a mass-insert or any get-locate from a program. If the operation is not explicitly ended by the program, CICS ends the operation at syncpoint or end of task.
- If there are more data buffers than strings, a start browse causes at least half the buffers to participate immediately in chained I/O. If the browse is short, the additional I/O is unnecessary.

Other VSAM definition parameters

Free space parameters need to be selected with care, and can help reduce the number of CI and CA splits. Where records are inserted all over a VSAM data set, it is appropriate to include free space in each CI. Where the inserts are clumped, free space in each CA is required. If all the inserts take place at just a few positions in the file, VSAM should be allowed to split the CA, and it is not necessary to specify any free space at all.

Adding records to the end of a VSAM data set does ***not*** cause CI/CA splits. Adding sequential records to anywhere but the end causes splits. An empty file with a low-value dummy key tends to reduce splits; a high-value key increases the number of splits.

VSAM resource usage (LSRPOOL)

The default for all VSAM data sets is LSR. If multiple pools are supported CICS provides for the use of pools 1 through 8

Effects

The LSRPOOLID parameter specifies whether a file is to use LSR or NSR and, if LSR, which pool.

Where useful

The LSRPOOLID parameter can be used in CICS systems with VSAM data sets.

Limitations

All files with the same base data set, except read-only files with DSNSHARING(MODIFYREQS) specified in the file definition, must use either the same LSR pool or all use NSR.

SERVREQ=REUSE files cannot use LSR.

Recommendations

See “VSAM considerations: general objectives” on page 221. Consider removing files from an LSR pool.

How implemented

The resource usage is defined by the LSRPOOL definition on the CSD. For more information about the CSD, see the *CICS Resource Definition Guide*.

VSAM buffer allocations for NSR (INDEXBUFFERS and DATABUFFERS)

For files using nonshared resources (NSR), the INDEXBUFFERS and DATABUFFERS parameters define VSAM index buffers and data buffers respectively.

Effects

INDEXBUFFERS and DATABUFFERS specify the number of index and data buffers for an NSR file.

The number of buffers can have a significant effect on performance. The use of many buffers can permit multiple concurrent operations (if there are the corresponding number of VSAM strings) and efficient sequential operations and CA splits. Providing extra buffers for high-level index records can reduce physical I/O operations.

Buffer allocations above the 16MB line represent a significant part of the virtual storage requirement of most CICS systems.

INDEXBUFFERS and DATABUFFERS have no effect if they are specified for files using LSR.

Where useful

The INDEXBUFFERS and DATABUFFERS parameters should be used in CICS systems that use VSAM NSR files in CICS file control.

Limitations

These parameters can be overridden by VSAM if they are insufficient for the strings specified for the VSAM data set. The maximum specification is 255. A specification greater than this will automatically be reduced to 255. Overriding of VSAM strings and buffers should never be done by specifying the AMP= attribute on the DD statement.

Recommendations

See "VSAM considerations: general objectives" on page 221.

How implemented

The INDEXBUFFERS and DATABUFFERS parameters are defined in the file definition on the CSD. They correspond exactly to VSAM ACB parameters: INDEXBUFFERS is the number of index buffers, DATABUFFERS is the number of data buffers.

For LSR files, they are ignored.

How monitored

The effects of these parameters can be monitored through transaction response times and data set and paging I/O rates. The CICS file statistics show data set activity to VSAM data sets. The VSAM catalog and RMF can show data set activity, I/O contention, space usage, and CI size.

VSAM buffer allocations for LSR

For files using local shared resources (LSR), the number of buffers to be used is not specified explicitly by file. The files share the buffers of the appropriate sizes in the LSR pool. The number of buffers in the pool may either be specified explicitly using the BUFFERS parameter in the file definition on the CSD, or be left to CICS to calculate. For more information about the CSD, see the *CICS Resource Definition Guide*.

Effects

The BUFFERS parameter allows for exact definition of specific buffers for the LSR pool.

The number of buffers can have a significant effect on performance. The use of many buffers can permit multiple concurrent operations (if there are the corresponding number of VSAM strings). It can also increase the chance of successful buffer lookaside with the resulting reduction in physical I/O operations.

The number of buffers should achieve an optimum between increasing the I/O saving due to lookaside and increasing the real storage requirement. This optimum is different for buffers used for indexes and buffers used for data. Note that the optimum buffer allocation for LSR is likely to be significantly less than the buffer allocation for the same files using NSR.

Where useful

The BUFFERS parameter should be used in CICS systems that use VSAM LSR files in CICS file control.

Recommendations

See "VSAM considerations: general objectives" on page 221.

How implemented

The BUFFERS parameter is defined in the file definition on the CSD. For more information about the CSD, see the *CICS Resource Definition Guide*.

How monitored

The effects of these parameters can be monitored through transaction response times and data set and paging I/O rates. The effectiveness affects both file and lsrpool statistics. The CICS file statistics show data set activity to VSAM data sets. The VSAM catalog and RMF can show data set activity, I/O contention, space usage, and CI size.

VSAM string settings for NSR (STRINGS)

STRINGS is used to determine the number of concurrent operations possible against the file and against the VSAM base cluster to which the file relates.

Effects

The STRINGS parameter for files using NSR has the following effects:

- It specifies the number of concurrent asynchronous requests that can be made against that specific file.
- It is used as the STRINGS in the VSAM ACB.
- It is used, in conjunction with the BASE parameter, to calculate the VSAM BSTRNO.
- A number greater than 1 can adversely affect performance for ESDS files used exclusively in write mode. With a string number greater than 1, the cost of invalidating the buffers for each of the strings is greater than waiting for the string, and there can be a significant increase in the number of VSAM EXCP requests.

Strings represent a significant part of the virtual storage requirement of most CICS systems. With CICS, this storage is above the 16MB line.

Where useful

The STRINGS parameter should be used in CICS systems that use VSAM NSR files in CICS file control.

Limitations

A maximum of 255 strings can be used as the STRNO or BSTRNO in the ACB.

Recommendations

See "Number of strings considerations for ESDS files" on page 225 and "VSAM considerations: general objectives" on page 221.

How implemented

The number of strings is defined by the STRINGS parameter in the CICS file definition on the CSD. It corresponds to the VSAM parameter in the ACB except where a base file is opened as the first for a VSAM data set; in this case, the CICS-accumulated BSTRNO value is used as the STRNO for the ACB.

How monitored

The effects of the STRINGS parameter can be seen in increased response times and monitored by the string queueing statistics for each file definition. RMF can show I/O contention in the DASD subsystem.

VSAM string settings for LSR (STRINGS)

STRINGS is used to determine the number of strings and thereby the number of concurrent operations possible against the LSR pool (assuming that there are buffers available).

Effects

The STRINGS parameter relating to files using LSR has the following effects:

- It specifies the number of concurrent requests that can be made against that specific file.
- It is used by CICS to calculate the number of strings and buffers for the LSR pool.
- It is used as the STRINGS for the VSAM LSR pool.
- It is used by CICS to limit requests to the pools to prevent a VSAM short-on-strings condition (note that CICS calculates the number of strings required per request).
- A number greater than 1 can adversely affect performance for ESDS files used exclusively in write mode. With a string number greater than 1, the cost of resolving exclusive control conflicts is greater than waiting for a string. Each time exclusive control is returned, a GETMAIN is issued for a message area, followed by a second call to VSAM to obtain the owner of the control interval.

Where useful

The STRINGS parameter can be used in CICS systems with VSAM data sets.

Limitations

A maximum of 255 strings is allowed per pool.

Recommendations

See "Number of strings considerations for ESDS files" on page 225 and "VSAM considerations: general objectives" on page 221.

How implemented

The number of strings is defined by the STRNO parameter in the file definition on the CSD, which limits the concurrent activity for that particular file.

How monitored

The effects of the STRINGS parameter can be seen in increased response times for each file entry. The CICS LSRPOOL statistics give information on the number of data set accesses and the highest number of requests for a string.

Examination of the string numbers in the CICS statistics shows that there is a two-level check on string numbers available: one at the data set level (see "File control" on page 379), and one at the shared resource pool level (see "LSRpool" on page 410).

RMF can show I/O contention in the DASD subsystem.

Maximum keylength for LSR (KEYLENGTH and MAXKEYLENGTH)

The KEYLENGTH, parameter in the file definition in the CSD, or the MAXKEYLENGTH in the LSR pool definition specifies the size of the largest key to be used in an LSR pool.

The maximum keylength may be specified explicitly using the KEYLENGTH parameter in the file definition on the CSD, or it may be left to CICS to determine from the VSAM catalog. For more information about the CSD, see the *CICS Resource Definition Guide*.

Effects

The KEYLENGTH parameter causes the "placeholder" control blocks to be built with space for the largest key that can be used with the LSR pool. If the KEYLENGTH specified is too small, it prevents requests for files that have a longer key length.

Where useful

The KEYLENGTH parameter can be used in CICS systems with VSAM data sets.

Recommendations

See "VSAM considerations: general objectives" on page 221.

The key length should always be as large as, or larger than, the largest key for files using the LSR pool.

How implemented

The size of the maximum keylength is defined in the KEYLEN parameter in the file definition on the CSD. For more information about the CSD, see the *CICS Resource Definition Guide*.

Resource percentile for LSR (SHARELIMIT)

The SHARELIMIT parameter in the LSR pool definition specifies the percentage of the buffers and strings that CICS should apply to the value that it calculates.

Effects

The method used by CICS to calculate LSR pool parameters and the use of the SHARELIMIT value is described in "VSAM considerations: general objectives" on page 221.

This parameter has no effect if both the BUFFERS and the STRINGS parameters are specified for the pool.

Where useful

The SHARELIMIT parameter can be used in CICS systems with VSAM data sets.

Recommendations

See "VSAM considerations: general objectives" on page 221.

Because SHARELIMIT can be applied only to files that are allocated at initialization of the LSR pool (when the first file in the pool is opened), it is always wise to specify the decimal STRINGS and BUFFERS for an LSR pool.

How implemented

The SHARELIMIT parameter is specified in the LSR pool definition. For more information, see the *CICS Resource Definition Guide*.

VSAM local shared resources (LSR)

Effects

CICS always builds a control block for LSR pool 1. CICS builds control blocks for other pools if either a LSR pool definition is installed, or a file definition at CICS initialization time has LSRPOOL= defined with the number of the pool.

Where useful

VSAM local shared resources can be used in CICS systems that use VSAM.

Recommendations

See "VSAM considerations: general objectives" on page 221.

How implemented

CICS uses the parameters provided in the LSR pool definition to build the LSR pool.

How monitored

VSAM LSR can be monitored by means of response times, paging rates, and CICS LSRPOOL statistics. The CICS LSRPOOL statistics show string usage, data set activity, and buffer lookasides (see "LSRpool" on page 410).

Hiperspace buffers

VSAM Hiperspace buffers reside in MVS expanded storage. These buffers are backed only by expanded storage. If the system determines that a particular page of this expanded storage is to be used for another purpose, the current page's contents are discarded rather than paged-out. If VSAM subsequently requires this page, it retrieves the data from DASD. VSAM manages the transfer of data between its Hiperspace buffers and its CICS address space buffers. CICS file control can only work with VSAM data when it is in a CICS address space buffer. Data is transferred between Hiperspace buffers and address space buffers in blocks of pages using CREAD and CWRITE commands. See "Hiperspace: data buffer statistics" on page 413 for more information.

Effects

The use of a very large number of Hiperspace buffers can reduce both physical I/O and pathlength when accessing your CICS files because the chance of finding the required records already in storage is relatively high.

Limitations

Because the amount of expanded storage is limited, it is possible that the installation will overcommit its use and VSAM may be unable to allocate all of the Hiperspace buffers requested. MVS may use expanded storage pages for purposes other than those allocated to VSAM Hiperspace buffers. In this case CICS continues processing using whatever buffers are available.

If address space buffers are similarly overallocated then the system would have to page. This overallocation of address space buffers is likely to seriously degrade CICS performance whereas overallocation of Hiperspace buffers is not.

Hiperspace buffer contents are lost when an address space is swapped out. This causes increased I/O activity when the address is swapped in again. If you use Hiperspace buffers, you should consider making the CICS address space nonswappable.

Recommendations

Keeping data in memory is usually very effective in reducing the CPU costs provided adequate central and expanded storage is available. Using mostly Hiperspace rather than all address space buffers can be the most effective option especially in environments where there are more pressing demands for central storage than VSAM data.

How implemented

CICS never requests Hiperspace buffers as a result of its own resource calculations. You have to specify the size and number of virtual buffers and Hiperspace buffers that you need.

You can use the RDO parameters of HSDATA and HSINDEX, which are added to the LSRPOOL definition to specify Hiperspace buffers. Using this method you can adjust the balance between Hiperspace buffers and virtual buffers for your system.

For further details of the CEDA transaction, see the *CICS Resource Definition Guide*.

Subtasking: VSAM (SUBTSKS=1)

Modes of TCB are as follows:

QR mode
There is always one quasi-reentrant mode TCB. It is used to run quasi-reentrant CICS code and non-threadsafe application code.

FO mode
There is always one file-owning TCB. It is used for opening and closing user datasets.

RO mode
There is always one resource-owning TCB. It is used for opening and closing CICS datasets, loading programs, issuing RACF calls, etc.

CO mode
The optional concurrent mode TCB is used for processes which can safely run in parallel with other CICS activity such as VSAM requests. The SIT keyword SUBTSKS has been defined to have numeric values (0 and 1) to specify whether there is to be a CO TCB.

SZ mode
The single optional SZ mode TCB is used by the FEPI interface.

RP mode
The single optional RP mode TCB is used to make ONC/RPC calls.

J8 mode
A task has a J8 mode TCB for its sole use if it needs to run a JVM.

L8 mode
L8 mode TCBs are not in use for CICS Transaction Server for OS/390 Release 3.

SO mode
The SO mode TCB is used to make calls to the sockets interface of TCP/IP.

SL mode
The SL mode TCB is used to wait for activity on a set of listening sockets.

S8 mode
A task has an S8 TCB for its sole use if it needs to use the system Secure Sockets Layer.

Effects

The objective of subtasks is to increase the maximum throughput of a single CICS system on multiprocessors. However, the intertask communication increases total processor utilization.

When I/O is done on subtasks, any extended response time which would cause the CICS region to stop, such as CI/CA splitting in NSR pools, causes only the additional TCB to stop. This may allow more throughput in a region that has very many CA splits in its file, but has to be assessed cautiously with regard to the extra overhead associated with using the subtask.

When the SUBTSKS=1 system initialization parameter has been specified:

- All Non-RLS VSAM file control WRITE requests to KSDS are subtasked.
- All other file control requests are never subtasked.

- Auxiliary temporary storage or intrapartition transient data requests are subtasked.
- Resource security checking requests are subtasked when the CICS main TCB (quasi-reentrant mode) exceeds approximately 70% activity.

Where useful

Subtasking can be useful with CICS systems that use VSAM.

Subtasking should only be used in a multiprocessing system in a region that is limited by a single processor but has spare capacity on other processors in the MVS image. If used in other circumstances, it can cause throughput degradation because of the dispatching of multiple tasks.

Limitations

Subtasking can improve throughput only in multiprocessor MVS images, because additional processor cycles are required to run the extra subtask. For that reason, we do not recommend the use of this facility on uniprocessors (UPs). It should be used only for a region that reaches the maximum capacity of one processor in a complex that has spare processor capacity or has NSR files that undergo frequent CI/CA splitting.

Regions that do not contain significant amounts of VSAM data set activity (particularly update activity) do not gain from VSAM subtasking.

Application task elapsed time may increase or decrease because of conflict between subtasking overheads and better use of multiprocessors. Task-related DSA occupancy increases or decreases proportionately.

Recommendations

SUBTSKS=1 should normally be specified only when the CICS system is run on a MVS image with two or more processors ***and*** the peak processor utilization due to the CICS main TCB in a region exceeds, say, about 70% of one processor, and a significant amount of I/O activity within the CICS address space is eligible for subtasking.

In this environment, the capacity of a second processor can be utilized to perform the I/O scheduling activity for VSAM data sets, auxiliary temporary storage, and intrapartition transient data.

The maximum system throughput of this sort of CICS region can be increased by using the I/O subtask, but at the expense of some additional processing for communication between the subtask and the MVS task under which the transaction processing is performed. This additional processing is seldom justified unless the CICS region has reached or is approaching its throughput limit.

A TOR that is largely or exclusively routing transactions to one or more AORs has very little I/O that is eligible for subtasking. It is not, therefore, a good candidate for subtasking.

An AOR is a good candidate only if a significant amount of VSAM I/O is performed within the AOR rather than being function-shipped to an FOR.

Subtasking should be considered for a busy FOR that often has a significant amount of VSAM I/O (but remember that DL/I processing of VSAM data sets is ***not*** subtasked).

How implemented

The system initialization parameter, SUBTSKS=1, defines that subtasking is to be used.

How monitored

CICS dispatcher domain statistics include information about the modes of TCB listed in "Subtasking: VSAM (SUBTSKS=1)" on page 237.

Note: CMF data and CICS trace are fully available.

Data tables

Data tables enable you to build, maintain and have rapid access to data records contained in tables held in virtual storage above the 16MB line. Therefore, they can provide a substantial performance benefit by reducing DASD I/O and pathlength resources. The pathlength to retrieve a record from a data table is significantly shorter than that to retrieve a record already in a VSAM buffer.

Effects

- After the initial data table load operation, DASD I/O can be eliminated for all user-maintained and for read-only CICS-maintained data tables.
- Reductions in DASD I/O for CICS-maintained data tables are dependent on the READ/WRITE ratio. This is a ratio of the number of READs to WRITEs that was experienced on the source data set, prior to the data table implementation. They also depend on the data table READ-hit ratio, that is, the number of READs that are satisfied by the table, compared with the number of requests that go against the source data set.
- CICS file control processor consumption can be reduced by up to 70%. This is dependent on the file design and activity, and is given here as a general guideline only. Actual results vary from installation to installation.

For CICS-maintained data tables, CICS ensures the synchronization of source data set and data table changes. When a file is recoverable, the necessary synchronization is already effected by the existing record locking. When the file is nonrecoverable, there is no CICS record locking and the note string position (NSP) mechanism is used instead for all update requests. This may have a small performance impact of additional VSAM ENDREQ requests in some instances.

Recommendations

- Remember that data tables are defined by two RDO parameters, TABLE and MAXNUMRECS of the file definition. No other changes are required.
- Start off gradually by selecting only one or two candidates. You may want to start with a CICS-maintained data table because this simplifies recovery considerations.
- Select a CICS-maintained data table with a high READ to WRITE ratio. This information can be found in the CICS LSRPOOL statistics (see page 410) by running a VSAM LISTCAT job.
- READ INTO is recommended, because READ SET incurs slightly more internal overhead.

- Monitor your real storage consumption. If your system is already real-storage constrained, having large data tables could increase your page-in rates. This in turn could adversely affect CICS system performance. Use your normal performance tools such as RMF to look at real storage and paging rates.
- Remember to select files that have a high proportion of full keyed direct reads as CICS-maintained data table candidates.
- Files that have a large proportion of update activity that does not require to be recovered across a restart would be better suited for user-maintained data tables.
- User-maintained data tables can use the global user exit XDTRD to modify as well as select records. This could allow the user-maintained data table to contain only the information relevant to the application.
- If storage isolation is specified allow for the extra storage needed by the data tables to prevent CICS incurring increased paging.

How implemented

Data tables can be defined using either the DEFINE FILE command of the CED*x* transaction or the DFHCSDUP utility program. See the *CICS Resource Definition Guide* for more information.

How monitored

Performance statistics are gathered to assess the effectiveness of the data table. They are in addition to those available through the standard CICS file statistics.

The following information is recorded:

- The number of attempts to read from the table
- The number of unsuccessful read attempts
- The number of bytes allocated to the data table
- The number of records loaded into the data table
- The number of attempts to add to the table
- The number of records rejected by a user exit when being added to the table either during loading or via the API
- The number of attempts to add a record which failed due to the table being full (already at its maximum number of records)
- The number of attempts to update table records via rewrite requests.
- The number of attempts to delete records from the table
- The highest value which the number of records in the table has reached since it was last opened.

There are circumstances in which apparent discrepancies in the statistics may be seen, caused, for example, by the existence of inflight updates.

Coupling facility data tables

For a description of how to define a coupling facility data table (CFDT), and start a coupling facility data table server, see the *CICS System Definition Guide*

A CFDT is similar in many ways to a shared user-maintained data table, and the API used to store and retrieve the data is based on the file control API used for user-maintained data tables. The data, unlike a UMT, is not kept in a dataspace in an MVS image and controlled by a CICS region, but kept in a coupling facility list structure, and control is shared between CFDT server regions. A CICS region

requesting access to a CFDT communicates with a CFDT server region running in the same MVS image, using the MVS authorised cross-memory (AXM) server environment. This is the same technique used by CICS temporary storage servers.

CFDTs are particularly useful for informal shared data. Uses could include a sysplex-wide shared scratchpad, look-up tables of telephone numbers, and creating a subset of customers from a customer list. Compared with existing methods of sharing data of this kind, such as shared data tables, shared temporary storage or RLS files, CFDTs offer some distinct advantages:

- If the data is frequently accessed for modification, CFDT provides superior performance compared with function-shipped UMT requests, or using an RLS file
- CFDT-held data can be recoverable within a CICS transaction. Recovery of the structure is not supported, but the CFDT server can recover from a unit of work failure, and in the event of a CICS region failure, a CFDT server failure, and an MVS failure (that is, updates made by units of work that were in-flight at the time of the failure are backed out). Such recoverability is not provided by shared temporary storage.

There are two models of coupling facility data table, a contention model or locking model.

Using the contention model, an exception condition (CHANGED) notifies an application that a rewrite following a read for update, or a delete following a read for update, needs to be retried because the copy of the record in the table has been updated by another task before the rewrite or delete could be performed. The contention model does not lock a record, but uses the version number of the table entry for the record to check that it has not been altered. If the version of this record on rewrite or delete is not the same as when the original read for update was performed, the CHANGED condition is returned.

The locking model causes records to be locked following a read for update request so that multiple updates cannot occur.

A contention model CFDT is non-recoverable. A locking model CFDT may be recoverable or non-recoverable. For a non-recoverable locking model, CFDT locks are held until a read for update sequence is completed by a rewrite or delete, but not until the next syncpoint. Changes are not backed out if a unit of work fails. In the recoverable case, locks are held until syncpoint, and the CFDT record is recoverable in the event of a unit of work failure or CICS region failure.

The relative cost of using update models and recovery is related to the number of coupling facility accesses needed to support a request. Contention requires the least number of accesses, but if the data is changed, additional programming and coupling facility accesses would be needed to handle this condition. Locking requires more coupling facility accesses, but does mean a request will not need to be retried, whereas retries can be required when using the contention model. Recovery also requires further coupling facility accesses, because the recovery data is kept in the coupling facility list structure.

The following table shows the number of coupling facility accesses needed to support the CFDT request types by update model.

Table 12. Coupling facility acess by request type and update model.

Request description	Contention	Locking	Recoverable
Open, Close	3	3	6
Read, Point	1	1	1
Write new record	1	1	2
Read for Update	1	2	2
Unlock	0	1	1
Rewrite	1	1	3
Delete	1	1	2
Delete by key	1	2	3
Syncpoint	0	0	3
Lock WAIT	0	2	2
Lock POST	0	2	2
Cross-system POST	0	2 per waiting server	2 per waiting server

Locking model

Records held in a coupling facility list structure are marked as locked by updating the adjunct area associated with the coupling facility list structure element that holds the data. Locking a record requires an additional coupling facility access to set the lock, having determined on the first access that the data was not already locked.

If, however, there is an update conflict, a number of extra coupling facility accesses are needed, as described in the following sequence of events:

1. The request that hits lock contention is initially rejected.
2. The requester modifies the locked record adjunct area to express an interest in it. This is a second extra coupling facility access for the lock waiter.
3. The lock owner has its update rejected because the record adjunct area has been modified, requiring the CICS region to re-read and retry the update. This results in two extra coupling facility accesses.
4. The lock owner sends a lock release notification message. If the lock was requested by a different server, this results in a coupling facility access to write a notification message to the other server and a coupling facility access to read it on the other side.

Contention model

The contention update model uses the entry version number to keep track of changes. The entry version number is changed each time the record is updated. This allows an update request to check that the record has not been altered since its copy of the record was acquired.

When an update conflict occurs, additional coupling facility accesses are needed:-

- The request that detects that the record has changed is initially rejected and a CHANGED response is sent.
- The application receiving the response has to decide whether to retry the request.

Effects

In a test that compared the use of a CFDT with a function-shipped UMT between 2 CICS regions running on different MVS members of a sysplex, it was found that overall CPU utilization was reduced by over 40% by using CFDTs. Some general observations that may be useful are:

- Access to CFDT records of 4094 bytes or less (4096 or 4K including 2 bytes of prefix data) are handled as synchronous coupling facility requests by the CFDT server. Requests for records of greater then 4K bytes are made asynchronously. These asynchronous accesses cost a little more in CPU usage and response time. In a benchmark test comparing the same transaction rates (337 per second) but different record sizes, the less-than-4K CFDT workload took 41.7% less CPU than the UMT equivalent. The greater than 4K CFDT workload took 41.1% less CPU with no measurable degradation of response time.
- Using the contention model requires the least coupling facility accesses but because the CHANGED condition needs to be handled and may need to be retried, maximum benefit is derived when there are few CHANGED conditions. These occurrences are reported in the CICS statistics which follow.
- If the CFDT records are 63 bytes or less in length, the record data is stored in the entry adjunct area of the coupling facility list structure, which gives improved performance when using the contention update mode.
- Using the locking model with recovery is the most costly mode of CFDT operation. Not only does this require more coupling facility accesses, but the CFDT server is also acting as a resource manager, co-ordinating the committal of updates in conjunction with the requesting CICS region. In a benchmark test involving the READ/UPDATE and REWRITE of CFDT records at a transaction rate of 168 per second, there was no significant difference in CPU utilization between transactions using contention and locking CFDTs. However, if the CFDT was defined as recoverable, the CPU utilization of the same transactions increased by approximately 15%.

Recommendations

Choose an appropriate use of a CFDT. For example, for cross-system, recoverable scratchpad storage, where shared TS does not give the required functional, or VSAM RLS incurs too much overhead.

A large file requires a large amount of coupling facility storage to contain it. Smaller files are better CFDT candidates (unless your application is written to control the number of records held in a CFDT).

The additional cost of using a locking model compared with a contention model is not great. Considering that using the contention model may need application changes if you are using an existing program, locking is probably the best choice of update model for your CFDT. If coupling facility accesses are critical to you, they are minimized by the contention model.

Recovery costs slightly more in CPU usage and in coupling facility utilisation.

Allow for expansion when sizing the CFDT. The amount of coupling facility storage a structure occupies can be increased dynamically up to the maximum defined in the associated coupling facility resource management (CFRM) policy with a SETXCF ALTER command. The MAXTABLES value defined to the CFDT server should allow for expansion. Therefore, consider setting it to a value higher than your initial requirements. If a CFDT does become full, its capacity can be increased using the CFDT operator command `SET TABLE=name,MAXRECS=n`.

The utilization of the CFDT should be regularly monitored both through CICS and CFDT statistics and RMF. Check that the size of the structure is reasonable for the amount of data it contains. A maximum-used of 80% is a reasonable target. Defining a maximum coupling facility list structure size in the CFRM policy definition to be greater than the initial allocation size specified by the POOLSIZE parameter in the CFDT server startup parameters enables you to enlarge the structure dynamically with a SETXCF ALTER command if the structure does fill in extraordinary circumstances.

Ensure that the AXMPGANY storage pool is large enough. This can be increased by increasing the REGION size for the CFDT server. Insufficient AXMPGANY storage may lead to 80A abends in the CFDT server.

How implemented

A CFDT is defined to a CICS region using a FILE definition with the following parameters:

- TABLE(CF)
- MAXNUMRECS(NOLIMIT | *number*(1 through 99999999))
- CFDTPOOL(*pool_name*)
- TABLENAME(*name*)
- UPDATEMODEL(CONTENTION | LOCKING)
- LOAD(NO³YES)

MAXNUMRECS specifies the maximum number of records that that CFDT can hold.

The first CICS region to open the CFDT determines the attributes for the file. Once opened successfully, these attributes remain associated with the CFDT through the data in the coupling facility list structure. Unless this table or coupling facility list structure is deleted or altered by a CFDT server operator command, the attributes persist even after CICS and CFDT server restarts. Other CICS regions attempting to open the CFDT must have a consistent definition of the CFDT, for example using the same update model.

The CFDT server controls the coupling facility list structure and the data tables held in this structure. The parameters documented in the *CICS System Definition Guide* describe how initial structure size, structure element size, and entry-to-element ratio can be specified.

How monitored

Both CICS and the CFDT server produce statistics records. These are described in "Appendix C. Coupling facility data tables server statistics" on page 503.

The CICS file statistics report the various requests by type issued against each CFDT. They also report if the CFDT becomes full, the highest number of records held and a Changed Response/Lock Wait count. This last item can be used to determine for a contention CFDT how many times the CHANGED condition was returned. For a locking CFDT this count reports how many times requests were made to wait because the requested record was already locked.

CFDT statistics

The CFDT server reports comprehensive statistics on both the coupling facility list structure it uses and the data tables it supports. It also reports on the storage used within the CFDT region by the AXM routines executed (the AXMPGLOW and

AXMPGANY areas). This data can be written to SMF and may also be produced automatically at regular intervals or by operator command to the joblog of the CFDT server.

The following is an example of coupling facility statistics produced by a CFDT server:

```
DFHCF0432I Table pool statistics for coupling facility list structure DFH
CFLS_PERFCFT2:
 Structure:    Size  Max size Elem size   Tables:   Current   Highest
             12288K    30208K       256                 4         4
 Lists:       Total    In use  Max used   Control      Data
                137        41        41        37         4
                100%       30%       30%       27%        3%
 Entries:     Total    In use  Max used      Free  Min free   Reserve
               3837      2010      2010      1827      1827       191
                100%       52%       52%       48%       48%        5%
 Elements:    Total    In use  Max used      Free  Min free   Reserve
              38691     12434     12434     26257     26257      1934
                100%       32%       32%       68%       68%        5%
```

This above example shows the amount of space currently used in a coupling facility list structure (Size) and the maximum size (Max size) defined for the structure. The structure size can be increased by using a SETXCF ALTER command. The number of lists defined is determined by the MAXTABLES parameter for the CFDT server. In this example, the structure can support up to 100 data tables (and 37 lists for control information).

Each list entry comprises a fixed length section for entry controls and a variable number of data elements. The size of these elements is fixed when the structure is first allocated in the coupling facility, and is specified to the CFDT server by the ELEMSIZE parameter. The allocation of coupling facility space between entry controls and elements will be altered automatically and dynamically by the CFDT server to improve space utilization if necessary.

The reserve space is used to ensure that rewrites and server internal operations can still function if a structure fills with user data.

The amount of storage used with the CFDT region to support AXM requests is also reported. For example:-

```
AXMPG0004I Usage statistics for storage page pool AXMPGANY:
     Size    In Use  Max Used      Free  Min Free
   30852K      636K      672K    30216K    30180K
     100%        2%        2%       98%       98%
                Gets     Frees   Retries     Fails
                3122      3098         0         0
AXMPG0004I Usage statistics for storage page pool AXMPGLOW:
     Size    In Use  Max Used      Free  Min Free
     440K       12K       12K      428K      428K
     100%        3%        3%       97%       97%
                Gets     Frees   Retries     Fails
                   3         0         0         0
```

The CFDT server uses storage in its own region for AXMPGANY and AXMPGLOW storage pools. AXMPGANY accounts for most of the available storage above 16MB in the CFDT region. The AXMPGLOW refers to 24-bit-addressed storage (below 16MB) and accounts for only 5% of this storage in the CFDT region. The CFDT server has a small requirement for such storage.

RMF reports

In addition to the statistics produced by CICS and the CFDT server, you can monitor the performance and use of the coupling facility list structure using the RMF facilities available on OS/390. A 'Coupling Facility Activity' report can be used to review the use of a coupling facility list structure. For example, this section of the report shows the DFHFCLS_PERFCFT2 structure size (12M), how much of the coupling facility is occupied (0.6%), some information on the requests handled, and how this structure has allocated and used the entries and data elements within this particular list structure.

```
                                          % OF             % OF     AVG    LST/DIR  DATA      LOCK     DIR REC/
      STRUCTURE                  ALLOC    CF         #     ALL      REQ/   ENTRIES  ELEMENTS  ENTRIES  DIR REC
TYPE  NAME              STATUS CHG SIZE     STORAGE    REQ   REQ      SEC    TOT/CUR  TOT/CUR   TOT/CUR  XI'S

LIST  DFHCFLS_PERFCFT2  ACTIVE     12M        0.6%    43530   93.2%    169.38   3837       39K      N/A       N/A
                                                                              1508       11K      N/A       N/A
```

RMF will also report on the activity (performance) of each structure, for example:-

```
 STRUCTURE NAME = DFHCFLS_PERFCFT2  TYPE = LIST
            # REQ    -------------- REQUESTS -------------   --------------- DELAYED REQUESTS -------------
SYSTEM     TOTAL             #     % OF  -SERV TIME(MIC)-    REASON    #   % OF  ---- AVG TIME(MIC) -----
 NAME       AVG/SEC           REQ    ALL     AVG   STD_DEV            REQ    REQ   /DEL    STD_DEV   /ALL

 MV2A       43530    SYNC     21K  49.3%   130.2     39.1
            169.4    ASYNC    22K  50.7%   632.7    377.7     NO SCH    0   0.0%    0.0        0.0    0.0
                     CHNGD     0    0.0%  INCLUDED IN ASYNC
                                                             DUMP      0   0.0%    0.0        0.0
```

This report shows how many requests were processed for the structure DFHFCLS_PERFCFT2 and average service times (response times) for the two categories of requests, synchronous and asynchronous. Be aware that requests of greater then 4K are handled asynchronously. For an asynchronous request, the CICS region can continue to execute other work and is informed when the request completes. CICS waits for a synchronous request to complete, but these are generally very short periods. The example above shows an average service time of 130.2 microseconds (millionths of a second). CICS monitoring records show delay time for a transaction due waiting for a CFDT response. In the example above, a mixed workload of small and large files was used. You can see from the SERV TIME values that, on average, the ASYNC requests took nearly 5 times longer to process and that there was a wide variation in service times for these requests. The STD_DEV value for SYNC requests is much smaller.

VSAM record-level sharing (RLS)

VSAM record-level sharing (RLS) is a VSAM data set access mode, introduced in DFSMS™ Version 1 Release 3, and supported by CICS. RLS enables VSAM data to be shared, with full update capability, between many applications running in many CICS regions. With RLS, CICS regions that share VSAM data sets can reside in one or more MVS images within an MVS parallel sysplex.

RLS also provides some benefits when data sets are being shared between CICS regions and batch jobs.

RLS involves the use of the following components:

- **A VSAM server, subsystem SMSVSAM**, which runs in its own address space to provide the RLS support required by CICS application owning regions (AORs), and batch jobs, within each MVS image in a Parallel Sysplex environment.

  The CICS interface with SMSVSAM is through an access control block (ACB), and CICS registers with this ACB to open the connection. Unlike the DB2 and DBCTL database manager subsystems, which require user action to open the

connections, if you specify RLS=YES as a system initialization parameter, CICS registers with the SMSVSAM control ACB automatically during CICS initialization.

A CICS region must open the control ACB to register with SMSVSAM before it can open any file ACBs in RLS mode. Normal file ACBs remain the interface for file access requests.

- **Sharing control data sets**. VSAM requires a number of these for RLS control. The VSAM sharing control data sets are logically-partitioned, linear data sets. They can be defined with secondary extents, but all the extents for each data set must be on the same volume.

  Define at least three sharing control data sets, for use as follows:

  – VSAM requires two active data sets for use in duplexing mode

  – VSAM requires the third data set as a spare in case one of the active data sets fails.

  See the *DFSMS/MVS DFSMSdfp Storage Administration Reference* for more information about sharing control data sets, and for a JCL example for defining them.

- **Common buffer pools and control blocks.** For data sets accessed in non-RLS mode, VSAM control blocks and buffers (local shared resources (LSR) pools) are located in each CICS address space and are thus not available to batch programs, and not even to another CICS region.

  With RLS, all the control blocks and buffers are allocated in an associated data space of the SMSVSAM server. This provides one extremely large buffer pool for each MVS image, which can be shared by all CICS regions connected to the SMSVSAM server, and also by batch programs. Buffers in this data space are created and freed automatically.

  DFSMS provides the RLS_MAX_POOL_SIZE parameter that you can specify in the IGDSMS*xx* SYS1.PARMLIB member. There are no other tuning parameters for RLS as there are with LSR pools—management of the RLS buffers is fully automatic.

Effects

There is an increase CPU costs when using RLS compared with function-shipping to an FOR using MRO. When measuring CPU usage using the standard DSW workload, the following comparisons were noted:

- Switching from local file access to function-shipping across MRO cross-memory (XM) connections incurred an increase of 7.02 ms per transaction in a single CPC.
- Switching from MRO XM to RLS incurred an increase of 8.20ms per transaction in a single CPC.
- Switching from XCF/MRO to RLS using two CPCs produced a *reduction* of 2.39ms per transaction.
- Switching from RLS using one CPC to RLS using two CPCs there was no appreciable difference.

In terms of response times, the performance measurements showed that:

- Function-shipping with MRO XM is better than RLS, but this restricts function-shipping to within one MVS image, and prevents full exploitation of a Parallel Sysplex with multiple MVS images or multiple CPCs.
- RLS is better than function-shipping with XCF/MRO, when the FOR is running in a different MVS image from the AOR.

However, performance measurements on their own don't tell the whole story, and do not take account of other factors, such as:

- As more and more applications need to share the same VSAM data, the load increases on the single file-owning region (FOR) to a point where the FOR can become a throughput bottleneck. The FOR is restricted, because of the CICS internal architecture, to the use of a single TCB for user tasks, which means that a CICS region generally does not exploit multiple CPs
- Session management becomes more difficult as more and more AORs connect to to the FOR.
- In some circumstances, high levels of activity can cause CI lock contention, causing transactions to wait for a lock even the specific record being accessed is not itself locked.

These negative aspects of using an FOR are resolved by using RLS, which provides the scalability lacking in a FOR.

How implemented

To use RLS access mode with CICS files:

1. Define the required sharing control data sets
2. Specify the RLS_MAX_POOL_SIZE parameter in the IGDSMS*xx* SYS1.PARMLIB member.
3. Ensure the SMSVSAM server is started in the MVS image in which you want RLS support.
4. Specify the system initialization parameter RLS=YES. This enables CICS to register automatically with the SMSVSAM server by opening the control ACB during CICS initialization. RLS support cannot be enabled dynamically later if you start CICS with RLS=NO.
5. Ensure that the data sets you plan to use in RLS-access mode are defined, using Access Method Services (AMS), with the required recovery attributes using the LOG and LOGSTREAMID parameters on the IDCAMS DEFINE statements. If you are going to use an existing data set that was defined without these attributes, redefine the data set with them specified.
6. Specify RLSACCESS(YES) on the file resource definition.

This chapter has covered the three different modes that CICS can use to access a VSAM file. These are non-shared resources (NSR) mode, local shared resources (LSR) mode, and record-level sharing (RLS) mode. (CICS does not support VSAM global shared resources (GSR) access mode.) The mode of access is not a property of the data set itself—it is a property of the way that the data set is opened. This means that a given data set can be opened by a user in NSR mode at one time, and RLS mode at another. The term non-RLS mode is used as a generic term to refer to the NSR or LSR access modes supported by CICS. Mixed-mode operation means a data set that is opened in RLS mode and a non-RLS mode concurrently, by different users.

Although data sets can be open in different modes at different times, all the data sets within a VSAM sphere must normally be opened in the same mode. (A sphere is the collection of all the components—the base, index, any alternate indexes and alternate index paths—associated with a given VSAM base data set.) However, VSAM does permit mixed-mode operations on a sphere by different applications, subject to some CICS restrictions.

How monitored

Using RLS-access mode for VSAM files involves SMSVSAM as well as the CICS region issuing the file control requests. This means monitoring the performance of both CICS and SMSVSAM to get the full picture, using a combination of CICS performance monitoring data and SMF Type 42 records written by SMSVSAM:

CICS monitoring

For RLS access, CICS writes performance class records to SMF containing:

- RLS CPU time on the SMSVSAM SRB
- RLS wait time.

SMSVSAM SMF data

SMSVSAM writes Type 42 records, subtypes 15, 16, 17, 18, and 19, providing information about coupling facility cache sets, structures, locking statistics, CPU usage, and so on. This information can be analyzed using RMF III post processing reports.

The following is an example of the JCL that you can use to obtain a report of SMSVSAM data:

```
//RMFCF     JOB (accounting_information),MSGCLASS=A,MSGLEVEL=(1,1),CLASS=A
//STEP1     EXEC PGM=IFASMFDP
//DUMPIN    DD DSN=SYS1.MV2A.MANA,DISP=SHR
//DUMPOUT   DD DSN=&&SMF,UNIT=SYSDA,
//          DISP=(NEW,PASS),SPACE=(CYL,(10,10))
//SYSPRINT  DD SYSOUT=*
//SYSIN     DD *
  INDD(DUMPIN,OPTIONS(DUMP))
  OUTDD(DUMPOUT,TYPE=000:255))
//POST      EXEC PGM=ERBRMFPP,REGION=0M
//MFPINPUT  DD DSN=&&SMF,DISP=(OLD,PASS)
//SYSUDUMP  DD SYSOUT=A
//SYSOUT    DD SYSOUT=A
//SYSPRINT  DD SYSOUT=A
//MFPMSGDS  DD SYSOUT=A
//SYSIN     DD *
 NOSUMMARY
 SYSRPTS(CF)
 SYSOUT(A)
 REPORTS(XCF)
/*
```

CICS file control statistics contain the usual information about the numbers of file control requests issued in the CICS region. They also identify which files are accessed in RLS mode and provide counts of RLS timeouts. They do not contain EXCP counts, ar any information about the SMSVSAM server, or its buffer usage, or its accesses to the coupling facility.

Chapter 20. Java program objects

This chapter describes CICS performance considerations for Java program objects built using VisualAge for Java, Enterprise Toolkit for OS/390 (ET/390). The following topics are included:

Overview

The high level of abstraction required for Java or any OO language involves increased layering and more dynamic runtime binding as a necessary part of the language. This incurs extra runtime performance cost.

The benefits of using Java language support include the ease of use of Object Oriented programming, and access to existing CICS applications and data from Java program objects. The cost of these benefits is currently runtime CPU and storage. Although there is a significant initialization cost, even for a Java program object built with ET/390, that cost amounts to only a few milliseconds of CPU time on the latest S/390® G5 processors. You should not see a noticeable increase in response time for a transaction written in Java unless CPU is constrained, although there will be a noticeable increase in CPU utilization. You can, however, take advantage of the scalability of the CICSplex architecture, and in particular, its parallel sysplex capabilities, to scale transaction rates.

When you execute a Java program object, a Language Environment run-unit or *enclave* is built and initialized for each invocation. You can reduce this performance overhead for frequently run Java program objects by requesting that a preinitialized and persistent enclave is reused for multiple invocations of the program.

This feature is known as **hot-pooling**.

Hot-pooling

CICS uses the PIPI preinitialization services of OS/390 Language Environment to build the run unit or enclave and executes the Java program object under control of a special class of Task Control Block (TCB) in the CICS region. Work done under these TCBs (known as H8 TCBs) runs concurrently with work done under the usual CICS quasi-reentrant TCB (QR), where most CICS transactions are executed.

For Java hot-pooling, Language Environment run-time options are defined in the user-supplied module DFHAPH8O. The sample module supplied by CICS contains the recommended values for STACK and HEAP sizes. Note that any values defined in the `-lerunopts` option of the **hpj** command are ignored for hot-pooling.

Note that the Language Environment storage HEAP grows as the preinitialized enclave is reused. When CICS determines that the available HEAP limit is approaching, the H8 TCB is terminated when the current program invocation completes. The next invocation of the program will require the re-establishment of

the enclave and TCB, with a performance impact. You can adjust the heap size to reduce the frequency of this performance impact, at the cost of higher storage usage.

You may use any number of hot-pooled enclaves, up to the limit defined by the MAXOPENTCBS system initialization parameter, but should note that each active enclave uses 20k bytes of below the line storage, if you use the default Language Environment run-time options supplied by the sample DFHAPH8O.

See the *CICS Transaction Server for OS/390 Release Guide* for more information about the Open Transaction Environment and the use of the QR TCB in CICS, and the *CICS Application Programming Guide* for information about using hot-pooling and the H8 TCBs. See also *OS/390 language Environment Programming Guide SC28–1939*for a description of the PIPI services.

Performance considerations

The main areas that may affect the CPU costs associated with running Java program objects with CICS, are discussed in the following sections:

- "DLL initialization"
- "LE runtime options"
- "API costs" on page 254
- "CICS system storage" on page 254

DLL initialization

At run time, when a Java program is initialized, all dynamic link libraries (DLLs) that contain functions that are referenced within that program are loaded into CICS storage. They remain in CICS storage until program compression occurs or they are explicitly refreshed using the CEMT SET NEWCOPY command. DLLs that have functions in them that are referenced by any of the DLLs being loaded are also brought into storage. This is referred to as 'aggressive loading'. Although the DLLs remain in storage, when they are reused by subsequent transactions, address resolution for all the functions within the DLLs is recalculated. Keeping the number of extraneous functions in diverse DLLs to a minimum can therefore reduce this initialization cost. Also, you should avoid grouping occasionally-used classes that refer to a diverse number of multiple DLLs with classes that are frequently used.

LE runtime options

Language environment (LE) runtime options can have a major impact on both storage usage and CPU costs of Java application programs running under CICS. The key LE runtime options for Java are STACK, HEAP and ANYHEAP. If the initial size for any of these options is too large, excessive storage will be allocated, which may result in a short-on-storage condition in the CICS region. If an initial value is too small, LE will issue a GETMAIN to allocate additional storage, which increases the CPU cost. Additional CPU cost can also be incurred due to extra GETMAINs and FREEMAINs if the FREE parameter is specified for any option where the initial size is too small.

LE runtime options for a Java program must be specified by the DFHAPH8O user-supplied module for hot-pooled programs, otherwise you can use the *-lerunopts* option of the **hpj** command, which is used to invoke the VisualAge® for Java, Enterprise Toolkit for OS/390 (ET/390) bytecode binder to bind Java bytecodes into a fully bound program object. For example,

```
-lerunopts="STACK(24K,4080,ANY,KEEP)"
```

The VisualAge for Java documentation is supplied in HTML format with the product. For more information about LE runtime options, see the *LE for OS/390 and VM Programming Reference* manual, (SC28–1940), and the *LE for OS/390 Customization* Guide, (SC28–1941).

To get a report on the storage used by your Java program object, specify the following runtime option on the **hpj** command

```
-lerunopts="RPTSTG(ON)"
```

When the Java program object is invoked in a CICS system, a storage report will be written to the CICS CESE transient data destination, which is usually directed to the data set defined by the CEEMSG DD statement. The report shows the number of system level get storage calls, such as EXEC CICS GETMAIN, that were required while the application was running. To improve performance, use the storage report numbers as an aid in setting the initial and increment size for STACK, HEAP and ANYHEAP to values which will reduce the number of times that the language environment storage manager makes requests to acquire storage. RPTSTG should only be used in a test environment because of the overheads incurred in writing the storage report each time the Java program object is executed.

Performance can also be improved by turning off the Java garbage collection routines. You do this by setting the following LE run time option:

```
'lerunopts="(envar('IBMHPJ_OPTS=-Xskipgc'))"
```

If you do not specify values for STACK, HEAP, and ANYHEAP when you use the **hpj** command to bind your Java program object, the program inherits default values from **hpj**. The defaults are:

STACK
: 128K, 4080, ANY, KEEP

HEAP 3200K, 300K, ANY, KEEP, 4K, 4080

ANYHEAP
: 20K, 4080, ANY, KEEP

The default values have been calculated to minimize excessive allocation of storage, but to provide sufficient storage so that the CPU cost is not pushed up by avoidable GETMAINs and FREEMAINs.

LE run-time options can only be explicitly specified for Java program objects which are built with the *-exe* option used on the **hpj** command. Program objects which are built with the *-jll* option, such as CICS CORBA server programs, inherit their LE runtime options from the invoking program. In the case of CICS CORBA server programs, this is DFJIIOP. The values used by DFJIIOP are:

STACK
: 24K, 4080, ANY, KEEP

HEAP 3200K, 300K, ANY, KEEP, 4K, 4080

ANYHEAP
20K, 4080, ANY, KEEP

API costs

When a Java program is executing it gains access to CICS resources via the JCICS classes. These classes 'wrap' a subset of the standard CICS API and give the Java application the ability to read a record from a VSAM KSDS file, for example. When accessing these CICS resources from a Java application there is an additional cost over and above that associated with the invocation of the API from other languages which use the CICS transalator. The costs associated with these various CICS APIs for the other languages are documented in "Appendix G. Performance data" on page 637. Although the additional cost with Java can vary slightly depending on the number of arguments passed, for the purposes of capacity planning and in keeping with the methodology stated in "Appendix G. Performance data" on page 637, you should add 6.5K instructions to any cost listed for the other languages.

CICS system storage

Java applications require more storage than programs written in traditional CICS programming languages, such as COBOL. Although a single Java application may run in a CICS system with an EDSA limit as low as 65M, you should specify a value much higher than this in the CICS EDSALIM system initialization parameter in order run a significant Java workload. The actual value you require for EDSALIM depends on the expected number of concurrently active Java program objects. Each invocation of a Java program object uses approximately 3.6M of EDSA.

Workload balancing of IIOP method call requests

As resource usage is so high when using Java program objects under CICS, it may be difficult to obtain the required throughput for your IIOP workload from one CICS region. Therefore, you may need to employ some form of workload balancing to spread the work over a number of cloned CICS regions.

There are three ways of balancing an IIOP workload over a number of CICS regions:

- CICS Dynamic Program Routing
- TCP/IP port sharing
- Dynamic Domain Name Server (DNS) registration for TCP/IP

CICS dynamic program routing

This is analogous to workload balancing using dynamic routing in a classic CICS TOR/AOR environment. Program DFJIIOP, the CICS ORB runtime which invokes the required CORBA method, is defined in the TOR as being dynamic. All IIOP requests are received by the TOR. Then, when program DFHIIOPA (the IIOP sender and application context handler) links to DFJIIOP, CICS dynamic routing services are used to invoke DFJIIOP in the least loaded or most efficient one of a series of cloned AORs. The AORs can be located anywhere in the sysplex.

TCP/IP port sharing

This provides a simple way of spreading the workload over multiple cloned CICS regions in one CEC by allowing all the CICS regions to listen on the same TCP/IP port number. The SHAREPORT parameter of the PORT TCP/IP configuration

statement is used to define the names of all of the CICS regions which may listen on a particular port (for more information, see the *OS/390 eNetworks Communications Server:IP Configuration* manual, SC31–8513). The IIOP TCPIPSERVICE definition installed in each CICS system has the same portnumber. When TCP/IP receives an incoming client connection request, it selects the listener (the active CICS region) with the fewest current connections (both active and in the backlog), and routes the request to that CICS region. All further requests from the same client are routed to the same CICS region until the connection is closed.

This method of workload balancing could be used in conjunction with CICS dynamic routing to provide multiple TORs in a TOR/AOR environment. This could eliminate a possible single point of failure.

Dynamic domain name server registration for TCP/IP

The dynamic domain name server (DNS) balances IP connections and workload in a Sysplex domain. The initial Interoperable Object reference (IOR) to the CICSplex contains a generic host name and port number. With dynamic DNS, multiple CICS systems are started to listen for IIOP requests on the same port (using virtual IP addresses), and the host name in the initial IOR is resolved to an IP address by MVS DNS and Workload Management (WLM) services.

Connection optimization in a sysplex domain is described in the *OS/390 eNetworks Communications Server: IP Planning and Migration Guide* (SC31–8512).

Chapter 21. Java virtual machine (JVM) programs

This chapter describes CICS performance considerations for Java programs run using the MVS Java Virtual Machine (JVM). The following topics are included:

Overview

Java application programs can be run under CICS control in CICS Transaction Server for OS/390 Release 3 and later releases, using the MVS Java Virtual Machine (JVM), which runs unchanged within CICS.

You can write CICS applications in Java and compile them to bytecode using any standard Java compiler, such as VisualAge for Java, or javac. Such programs will be referred to as JVM programs in order to distinguish them from Java Program Objects that are built using VisualAge for Java, Enterprise Toolkit for OS/390.

When a JVM program executes, an MVS JVM running inside CICS is interpreting the Java bytecodes. When a Java Program Object executes, it is running OS/390 machine code with runtime support from Language Environment (LE/370).

Java Program Objects are restricted to a subset of the core Java classes whereas a JVM program can use the full Java package set.

Performance considerations

JVM programs are not recommended to be used for high volume, high priority transactions. Java Program Objects provide better performance in the CICS environment. Indeed it is expected that the vast majority of Java programs running in CICS will be Java Program Objects and that the restrictions concerning which core classes can be used by Java Program Objects will not be an issue for most programs.

JVM programs execute by means of the MVS JVM interpreting the Java bytecodes. This interpretation will involve more CPU usage than for conventionally compiled programs executing platform—specific machine code. JVM programs cannot take advantage of the JVM Just-in-time (JIT) compiler because at present CICS cannot safely reuse a JVM created for one JVM program in order to run a second JVM program. A JVM is created and destroyed for each JVM program that is run. It is recommended that the JVM is run with the JIT disabled. This is the default setting shipped by CICS in the DFHJVMEV member of the SDFHENV dataset which contains the JVM tailorable options.

A large part of the CPU overhead associated with running JVM programs is consumed with creating and destroying a JVM each time a JVM program is run. As JVM technology evolves over time, it is anticipated that CICS will be able to reuse JVMs, thereby allowing costs to be lowered.

A Java Virtual Machine executing a JVM program is run inside CICS under its own TCB. The CICS-JVM Interface uses the Open Transaction Environment (OTE)

function provided in CICS Transaction Server for OS/390 Release 3 to provide the "Open TCB" under which the JVM is run. Each JVM program running in CICS is running its own JVM on its own open TCB. The particular type of open TCB provided for use by the JVM is called a J8 TCB. The priority of J8 TCBs is set significantly lower than that of the main CICS QR TCB to ensure that JVM programs, which have a high CPU cost, are treated as low priority transactions and so do not affect the main CICS workload being processed under the CICS QR TCB.

Storage usage

An MVS JVM runs under a J8 TCB within the CICS address space. It runs as a Unix System Services process and utilizes MVS Language Environment® services rather than CICS Language Environment services. That is to say, it uses the variant of Language Environment (LE) normally used outside of CICS, for example, by a batch COBOL program. The JVM uses MVS LE and not CICS LE because CICS LE—conforming applications do not support threading. As a result, all storage obtained by the JVM is MVS storage, acquired with a GETMAIN, within the CICS region, but outside of the CICS DSAs.

An MVS JVM uses a significant amount of storage in the region. Multiple JVM instances running within CICS will require a significant increase in the CICS region size. Each JVM requires the following:

- A minimum of 112K storage below the line. If the Java application utilises Java threads, there is an additional 5-6K of storage consumed for each thread.
- A minimum of 19M of storage above the 16MB line. This very substantial amount of storage includes stack and heap storage used to create Java objects. The figure is based upon running the JVM with the IBM recommended values for stack and heap as shipped in the DFHJVMEV member of the SDFHENV dataset.

The amount of storage required above the 16MB line by the JVM means that a minimum region size of 40M is required to run a single JVM inside CICS. Allowing for multiple JVM instances to run inside CICS will require a significant increase in region size. This may require changes to installation exits, IEALIMIT or IEFUSI, that are used to limit the region size of a job. Note that running with a default IEFUSI and specifying REGION=0M will result in a region size of 32M which is not enough to support a JVM.

The amount of storage required below the line by the JVM effectively puts a maximum limit of 30 JVMs in a CICS address space, assuming a DSALIMIT of 4M and assuming enough storage above the line is available. It is recommended that transactions running JVM programs are limited using techniques such as TRANCLASS. An alternative approach could be to have a JOR, a JVM owning region, to which all JVM program executions are routed. Such a region would run only JVM workloads thereby minimising the amount of CICS DSA storage required and allowing the maximum amount of MVS storage to be allocated for use by JVMs.

How monitored

The following facilities are available to monitor the CICS-JVM interface:

- The CICS monitoring facility may be used to monitor the CPU time used by a transaction that invokes a JVM program including the amount of CPU time used by the MVS JVM on the J8 TCB. The CICS monitoring facility also includes the elapsed time spent in the JVM and the number of JCICS API requests issued by the JVM program.

- The CICS sample DFH0STAT provides statistical information including the amount of user storage allocated above and below the 16MB line. The same information can also be obtained from the job termination message IEF374I. 'VIRT=nnnnnK' shows you the virtual storage below the 16MB line, and 'EXT=nnnnnnnK ' shows you the virtual storage above the 16MB line.

Chapter 22. Database management

This chapter includes the following topics:

DBCTL minimum threads (MINTHRD)

This parameter specifies the number of threads that are created when CICS connects to DBCTL. They remain allocated while the database resource adapter (DRA) is active. These threads continue to remain allocated until the CICS system is disconnected from DBCTL, unless a thread is stopped by a /STOP command or by a thread failure.

Effects

The DRA allocates control blocks for the specified number of threads at DBCTL connection time. One thread is equivalent to one MVS TCB, thus giving more concurrency on multiprocessors. Because these threads are available for the duration of the DBCTL connection, there is no pathlength overhead for collapsing and reallocating thread related storage, and throughput should, therefore, be faster.

The number you specify should be large enough to cover average DL/I transaction loads. After the MINTHRD limit is reached, additional threads are allocated up to the MAXTHRD limit, the number specified in the MAXREGN, or the maximum of 255, whichever is the lowest.

When multiple CICS systems or Batch message processing programs (BMPs) are connected to DBCTL, the sum of MINTHRD and BMPs must be less than or equal to MAXREGN (MAXREGN is specified in the IMS sysgen macros).

Where useful

MINTHRD can be used in DBCTL systems to synchronize thread allocation with workload requirements.

Limitations

There is a storage allocation of about 9KB per thread in the local system queue area (LSQA) below the 16MB line.

Implementation

The MINTHRD and MAXTHRD parameters are specified in the DRA startup table (DFSPZP).

How monitored

DBCTL statistics are available when the CICS/DBCTL interface is shut down normally. The MINTHRD value is recorded (see page 358 for further information). You can also use CICS auxiliary trace to check for queueing for threads and PSBs.

DBCTL maximum threads (MAXTHRD)

The MAXTHRD parameter specifies the maximum number of threads that this CICS system may use up to a value of 255, or the limit imposed by MAXREGN. The default is 1 or the number defined by MINTHRD, whichever is the highest.

Effects

This parameter controls the maximum number of tasks for which this CICS system can have PSBs scheduled in DBCTL. Any requests to schedule a PSB when the MAXTHRD limit is reached is queued by the DRA.

Where useful

MAXTHRD can be used in DBCTL systems to ensure that, at peak loads, additional threads can be built in addition to those already allocated as a result of MINTHRD, thus avoiding waiting for threads.

Limitations

After the MINTHRD limit is exceeded, threads continue to be built up to the MAXTHRD limit but, because each thread's control blocks are allocated during PSB scheduling, the pathlength is greater for the tasks running after the MINTHRD limit has been reached.

Implementation

The MINTHRD and MAXTHRD parameters are specified in the DRA startup table (DFSPZP).

How monitored

DBCTL statistics are available when the CICS/DBCTL interface is shut down normally. The MAXTHRD value is recorded (see page 358 for further information). You can also use CICS auxiliary trace to check for queueing for threads and PSBs.

DBCTL DEDB parameters (CNBA, FPBUF, FPBOF)

Because DEDB parameters are defined both in the CICS region and the IMS/ESA (DBCTL) region, both sets of interdependent parameters are included here.

If you use DEDBs, you must define the characteristics and usage of the IMS/ESA DEDB buffer pool. You do this by specifying parameters (including DRA startup parameters) during IMS/ESA system definition or execution.

The main concerns in defining DEDB buffer pools are the total number of buffers in the IMS/ESA region, and how they are shared by CICS threads. You use the following IMS/ESA FPCTRL parameters to define the number of buffers:

- DBBF: total number of buffers
- DBFX: number of buffers used exclusively by the DEDB system.

The number remaining when you subtract the value specified for DBFX from the value specified for DBBF is the number of buffers available for the needs of CICS threads. In this discussion, we have assumed a fixed number for DBFX. DBBF must, therefore, be large enough to accommodate all batch message processing programs (BMPs) and CICS systems that you want to connect to this DBCTL system.

When a CICS thread connects to IMS/ESA, its DEDB buffer requirements are specified using a normal buffer allocation (NBA) parameter. For a CICS system, there are two NBA parameters in the DRA startup table:

1. CNBA buffers needed for the CICS system. This is taken from the total specified in DBBF.
2. FPBUF buffers to be given to each CICS thread. This is taken from the number specified in CNBA. FPBUF is used for each thread that requests DEDB resources, and so should be large enough to handle the requirements of any application that can run in the CICS system.

A CICS system may fail to connect to DBCTL if its CNBA value is more than that available from DBBF. An application may receive schedule failure if the FPBUF value is more than that available from CNBA. The FPBUF value is used when an application tries to schedule a PSB that contains DEDBs.

When a CICS system has connected to DBCTL successfully, and the application has successfully scheduled a PSB containing DEDBs, the DRA startup parameter FPBOF becomes relevant. FPBOF specifies the number of overflow buffers each thread gets if it exceeds FPBUF. These buffers are not taken from CNBA. Instead, they are buffers that are *serially* shared by all CICS applications or other dependent regions that are currently exceeding their normal buffer allocation (NBA) allocation.

Because overflow buffer allocation (OBA) usage is serialized, thread performance can be affected by NBA and OBA specifications. If FPBUF is too small, more applications need to use OBA, which may cause delays due to contention. If both NBA and OBA are too small, the application fails. If FPBUF is too large, this affects the number of threads that can concurrently access DEDB resources, and increases the number of schedule failures.

Where useful

The DBCTL DEDB parameters are useful in tuning a CICS/DBCTL DEDB fastpath environment.

Recommendations

In a CICS/DBCTL environment, the main performance concern is the trade-off between speed and concurrency. The size of this trade-off is dictated by the kind of applications you are running in the CICS system.

If the applications have approximately the same NBA requirements, there is no trade-off. You can specify an FPBUF large enough to never need OBA. This speeds up access and there is no waste of buffers in CNBA, thus enabling a larger number of concurrent threads using DEDBs.

The more the buffer requirements of your applications vary, the greater the trade-off. If you want to maintain speed of access (because OBAs are not being used) but decrease concurrency, you should increase the value of FPBUF. If you

prefer to maintain concurrency, do not increase the value of FPBUF. However, speed of access decreases because this and possibly other threads may need to use the OBA function.

For further guidance on DEDB buffer specification and tuning, see the information on DEDBs in the *IMS/ESA Database Administration Guide*, and the *IMS/ESA System Administration Guide*.

How implemented

DBBF and DBFX are parameters defined during DBCTL system generation or at DBCTL initialization. CNBA, FPBUF, and FPBOF are defined in the DRA startup table (DFSPZP).

How monitored

Monitoring data at the transaction level is returned to CICS by DBCTL at schedule end and transaction termination. This data includes information on DEDB statistics.

Note: To obtain the monitoring data, two event monitoring points (EMPs) must be added to your CICS monitoring control table (MCT). For information about coding the DBCTL EMPs, see the *CICS Customization Guide*.

CICS DB2 attachment facility

The CICS DB2 attachment facility provides a multi-thread connection to DB2. The connections between CICS and DB2 are called threads. There are three types of thread:

Command threads.
One or more threads can be reserved for command usage. It is used for DB2 commands only; for example, the DSNC -DIS command. During periods of heavy command usage, requests for command threads may be transferred to pool threads.

Pool threads
Pool threads are normally used for low volume transactions, and transactions that overflow from either entry threads or command threads. These threads are created when needed and terminate immediately when unused.

Entry threads
These threads are intended for high-volume, high-priority transactions. Each thread is associated with a particular application plan, and the threads are reusable. Entry threads can be defined as *protected* which means that they are not terminated immediately if unused. They are terminated after two consecutive periods of inactivity. These periods are defined in the DB2CONN parameter, PURGECYCLE. Many CICS transactions can use the same protected threads and avoid the overhead involved in creating and terminating the thread for each transaction.

The DB2CONN, DB2ENTRY, and DB2TRAN definitions of the CICS DB2 attachment facility define the authorization and access attributes on a transaction and transaction group basis.

Effects

The THREADWAIT parameter of DB2CONN, and DB2ENTRY define whether the requests for a thread should be queued, abended, or sent to the pool thread in the case of a shortage of entry or command threads. If THREADWAIT=YES is specified instead of THREADWAIT=POOL the transaction is queued rather than sent to the pool thread. Using THREADWAIT=YES, therefore, avoids the thread initialization and termination overhead. If a transaction is made to wait because of the lack of entry threads, a queueing arrangement is necessary. This is done by the CICS DB2 attachment facility. The advantages of this are that, once the entry thread finishes its current piece of work, it continues with the next transaction immediately.

You can optimize performance between CICS and DB2 by adjusting the transaction class limits, MXT system parameters of CICS and the THREADWAIT, TCBLIMIT, THREADLIMIT, and PRIORITY attributes of DB2CONN, and DB2ENTRY.

.

Where useful

In a high-volume, highly-utilized system using DB2.

How implemented

THREADWAIT is defined in the DB2CONN and DB2ENTRY definitions of the CICS DB2 attachment facility.

How monitored

The following facilities are available to monitor the CICS DB2 attachment facility.

- The CICS auxiliary trace facility and the CICS monitoring facility may be used to trace and monitor the SQL calls issued by a specific CICS application program.
- The CICS DB2 attachment facility command (DSNC DISPLAY) provides information about CICS transactions accessing DB2 data, or statistical information associated with entries in resource definition online.
- There are also various DB2 facilities which can be used. (See the *DB2 Administration Guide* for more information.)

The CICS performance class monitoring records include the following DB2–related data fields:

- The total number of DB2 EXEC SQL and instrumentation facility interface (IFI) requests issued by a transaction.
- The elapsed time the transaction waited for a DB2 thread to become available.
- The elapsed time the transaction waited for a CICS DB2 subtask to become available.
- The elapsed time the transaction waited for DB2 to service the DB2 requests issued by the transaction.

CICS monitoring is used in the CICS DB2 environment with the DB2 accounting facility, to monitor performance and to collect accounting information.

CICS DB2 attachment facility (TCBLIMIT, and THREADLIMIT)

TCBLIMIT, and THREADLIMIT are parameters in the DB2CONN and DB2ENTRY definitions. They can be set for each of the three types of thread (see page 264 for more information). TCBLIMIT specifies the maximum number of threads between CICS and DB2, which, in turn, specifies the maximum number of active DB2 threads. THREADLIMIT specifies the maximum number of active DB2 threads. THREADLIMIT is modified dynamically.

The sum of all the active threads from TSO users, all CICS and IMS systems and other systems accessing DB2 should not exceed CTHREAD. Otherwise, the result could be unpredictable response times. When this occurs, a CICS DB2 attachment facility "create thread" request is queued by DB2, and the CICS transaction is placed is a wait state until a thread is available.

Note: CTHREAD is a DB2 parameter, specified in ZPARMS, and it defines the number of concurrent threads for all of DB2.

Effect

Each thread linking CICS to DB2 has a corresponding TCB in the CICS address space. Too many TCBs per address space involve the MVS dispatcher scanning the TCBs to identify an active TCB. If there is a large number of TCBs then there may be a significant cost of processor time.

Limitations

Increasing the TCBLIMIT value or setting up an additional CICS system with access to the same DB2 system may require increasing the CTHREAD parameter of DB2.

Recommendations

For a protected entry thread environment, implementation involves reviewing the number of application plans and, if possible, reducing the number of plans by combining infrequently used ones while balancing the issues of plan size and security.

Initially, you should start with one thread per plan. In a high-volume transaction processing environment, you can estimate the initial number by using the occupancy time of a thread by a transaction and multiplying it with the expected transaction rate. For example, an occupancy time of 0.2 seconds and a transaction rate of 20 transactions per second (0.2 x 20) would give an initial thread number of between three and four.

How monitored

The following facilities are available to monitor the CICS DB2 attachment facility.

- The CICS auxiliary trace facility and the CICS monitoring facility may be used to trace and monitor the SQL calls issued by a specific CICS application program.
- The CICS DB2 attachment facility command (DSNC DISPLAY) provides information about CICS transactions accessing DB2 data, or statistical information associated with attributes of DB2CONN, and DB2ENTRY.
- There are also various DB2 facilities which can be used. (See the *DB2 Administration Guide* for more information.)

CICS DB2 attachment facility (PRIORITY)

PRIORITY is a parameter of the DB2CONN and DB2ENTRY definitions of the CICS attachment facility that can be specified for both the pool and entry threads. This parameter controls the priority of the CICS TCBs. There are three options: PRIORITY=HIGH, PRIORITY=LOW, and PRIORITY=EQUAL. (See the *DB2 Administration Guide* for more information.)

Effects

When PRIORITY=HIGH is specified, transactions run at a higher priority than CICS thus saving virtual storage, releasing locks, and avoiding other transactions deadlocking or timing out. However, if all threads are specified with PRIORITY=HIGH, CICS itself may be effectively at too low a priority.

Where useful

Setting PRIORITY=HIGH is useful for high-priority and high-volume transactions.

Limitations

A complex SQL call could spend a long time in DB2, and the CICS TCB may not be dispatched.

Recommendations

Set PRIORITY=HIGH for your transactions with the highest weighted average number of SQL calls. The highest weighted average is equal to the number of SQL calls per transaction multiplied by the frequency of transaction. Set PRIORITY=LOW or EQUAL for other transactions. If the CPU usage per call is high, you should not set PRIORITY=HIGH.

How implemented

PRIORITY is a parameter of the DB2CONN and DB2ENTRY definitions of the CICS attachment facility.

How monitored

The following facilities are available to monitor CICS attachment facility.

- The CICS auxiliary trace facility and the CICS monitoring facility may be used to trace and monitor the SQL calls issued by a specific CICS application program.
- The CICS attachment facility command (DSNC DISPLAY) provides information about CICS transactions accessing DB2 data, or statistical information associated with DB2 resource definitions in the CSD.
- There are also various DB2 facilities which can be used. (See the *DB2 Administration Guide* for more information.)

Chapter 23. Logging and journaling

This chapter discusses performance tuning issues related to logging and journaling:

Coupling facility or DASD-only logging?

In earlier releases of MVS before OS/390 Version 2 Release 5, the MVS system logger requires at least one coupling facility, even if the sysplex consists of a single MVS image; all CICS log streams must use log structures defined in the coupling facility.

The CICS log manager supports the DASD-only option of the MVS system logger. This means that individual CICS log streams can use either coupling facility log structures or DASD-only logging. (For more information about the types of storage used by CICS log streams, see the *CICS Transaction Server for OS/390 Installation Guide*.)

If you have a coupling facility, the *CICS Transaction Server for OS/390 Installation Guide* contains advice on how you could define each log stream, based on its usage. For information about the relative performance of CF and DASD-only log streams, see Table 197 on page 639.

Integrated coupling migration facility

If you use a coupling facility, you can use a stand-alone model, such as the S/390 9674. Alternatively, you can use the integrated coupling migration facility (ICMF) to provide the services of a coupling facility in an LPAR. This means that the coupling facility and MVS are not failure-independent, thereby requiring the use of staging data sets.

Monitoring the logger environment

CICS collects statistics on the data written to each journal and log stream; this data can be used to analyze the activity of a single region. However, because general log streams can be shared across multiple MVS images, it can be more useful to examine the statistics generated by MVS.

The MVS system logger writes SMF Type 88 records containing statistics for each connected log stream. MVS supplies in SYS1.SAMPLIB a sample reporting program, IXGRPT1, that you can use as supplied, or modify to meet your requirements. Alternatively, you could use some other SMF reporting program. For

information about the SMF Type 88 records and the sample reporting program, see the *OS/390 MVS System Management Facilities (SMF)* manual.

The main items that should be monitored routinely are:
- For coupling facility log streams, the number of "structure full" events
- For DASD-only log streams, the number of "staging data set full" events.

If these events occur frequently, this indicates that the logger cannot write data to secondary storage quickly enough to keep up with incoming data, which causes CICS to wait before it can write more data. Consider the following solutions to resolve such problems:

- Increase the size of primary storage (that is, the size of the coupling facility structure or, for a DASD-only log stream, the size of the staging data set), in order to smooth out spikes in logger load.
- Reduce the data written to the log stream by not merging so many journals or forward recovery logs onto the same stream.
- Reduce the HIGHOFFLOAD threshold percentage, the point at which the system logger begins offloading data from primary storage to offload data sets.
- Review the size of the offload data sets. These should be large enough to avoid too many "DASD shifts"—that is, new data set allocations. Aim for no more than one DASD shift per hour. You can monitor the number of DASD shifts using the SMF88EDS record.
- Examine device I/O statistics for possible contention on the I/O subsystem used for offload data sets.
- Use faster DASD devices

For CICS system logs, the best performance is achieved when CICS can delete log tail data that is no longer needed before it is written to secondary storage by the MVS system logger. To monitor that this is being achieved, your reporting program should examine the numbers in the SMF88SIB and SMF88SAB SMF Type 88 records. These values indicate:

SMF88SIB
: Data deleted from primary storage without first being written to DASD offload data sets. For a system log stream, this value should be high in relation to the value of SMF88SAB. For a general log stream, this value should normally be zero.

SMF88SAB
: Data deleted from primary storage *after* being written to DASD offload data sets. For a system log stream, this value should be low in relation to the value of SMF88SIB. For a general log stream, this value should normally be high.

Note: In any SMF interval, the total number of bytes deleted from primary storage (SMF88SIB plus SMF88SAB) may not match the total number of bytes written to secondary storage, because data is only written to offload data sets and then deleted from primary storage when the HIGHOFFLOAD threshold limit is reached.

If the SMF88SAB record frequently contains high values for a CICS system log:

- Check that RETPD=dddd is not specified on the MVS definition of the logstream. (For information about the MVS RETPD parameter, see the *CICS Transaction Server for OS/390 Installation Guide*.)

- Check that no long-running transactions are making recoverable updates without syncpointing.
- Consider increasing the size of primary storage.
- Consider increasing the HIGHOFFLOAD threshold value.
- Consider reducing the value of the the AKPFREQ system initialization parameter.

Average blocksize

Important
This section applies only to log streams that use coupling facility structures.

Although consideration of the average blocksize written to the coupling facility can happen only at the level of application design, it is still worth bearing in mind when considering the performance implications of the CICS log manager.

If the average blocksize of data being written to the coupling facility is less than 4K, the write request is processed synchronously. Not only is the operation synchronous to CICS, but the System/390® instruction used to access the coupling facility is also synchronous, in that it executes for as long as it takes to place the data in the structure. For this reason, it is unwise to mix fast CPUs with slow coupling facilities. If the access time to a particular coupling facility remains constant, then, for synchronous accesses, the faster the CPU the more CPU cycles are consumed by the request.

If the average blocksize of data being written to the coupling facility is greater than 4K bytes, the write request is processed asynchronously; the CICS task gives up control and the MVS system logger posts the ECB when the write request has been satisfied. This can result in an asynchronous request taking longer to complete than a synchronous one. However, there is no System/390 "long instruction" to place data into the coupling facility.

Synchronous requests may be changed into asynchronous requests, if the sub-system decides this to be necessary—for example, if the sub-channel is busy. Changed requests appear on an RMF III report as CHNGD. Figure 32 shows an extract from an RMF report showing the numbers of synchronous and asynchronous writes to a coupling facility structure.

```
STRUCTURE NAME = LOG_FV_001        TYPE = LIST
          # REQ    -------------- REQUESTS -------------
SYSTEM    TOTAL             #     % OF  -SERV TIME(MIC)-
NAME      AVG/SEC           REQ    ALL     AVG   STD_DEV

MV2A      15549    SYNC     15K  95.3%   476.1    339.6
          27.87    ASYNC   721    4.6%  3839.0   1307.3
                   CHNGD    12    0.1%  INCLUDED IN ASYNC
```

Figure 32. RMF report showing numbers of synchronous and asyncronous writes to a coupling facility

Number of log streams in the CF structure

Important
This section applies only to log streams that use coupling facility structures.

Coupling facility space is divided into structures by the CFRM policy, the maximum permitted being 255 structures. Multiple log streams can use the same structure. Generally, the more log streams per structure, the more difficult it is to tune the various parameters that affect the efficiency and performance of the CICS log manager.

AVGBUFSIZE and MAXBUFSIZE parameters

OS/390 Release 3 and later
If you are running on OS/390 Release 3 or later, the value you specify for AVGBUFSIZE is less important than if you are running on an earlier release of MVS. This is because OS/390 Release 3 and later dynamically tune the element/entry ratio described in this section.

As far as performance considerations go, you should try to ensure that log streams used by applications that write similar sized data records share the same structure. The reasons for this relate to the values defined in the AVGBUFSIZE and MAXBUFSIZE parameters on the structure definition.

When a CF structure is defined, it is divided into two areas:

- One area holds list entries
- The other area holds list elements.

List elements are units of logged data and are either 256 bytes or 512 bytes long. List entries are index pointers to the list elements. There is one list entry per log record. There is at least one element per log record.

If you define MAXBUFSIZE with a value greater than 65276, data is written in 512-byte elements. If you define MAXBUFSIZE with a value less than, or equal to, 65276, data is written in 256-byte elements. The maximum value permitted for this parameter is 65532.

The proportion of the areas occupied by the list entries and the list elements is determined by a ratio calculated as follows:

```
AVGBUFSIZE / element size
```

The resulting ratio represents the ratio, *nn* : 1, where *nn* represents element storage, and '1' represents entry storage. This is subject to a minimum of 1:1.

This ratio has performance significance because it may be inappropriate for a combination of many different applications with different logging requirements and behavior.

Element/entry ratio and the number of log streams per structure

AVGBUFSIZE is set at the structure level and dictates the ratio for the whole structure. As a general rule, the greater the number of log streams per structure,

the greater the chance that the element/entry ratio is inappropriate for certain applications using the log streams. If many applications write to their log streams significantly differing amounts of data, and at significantly differing intervals, some of those applications can experience unexpected DASD offloading, with the extra processing overhead that this incurs. The DASD offloading is unexpected because the log stream may not yet have reached its HIGHOFFLOAD threshold.

Each log record places an entry in the list entry area of the structure, and the data is loaded as one or more elements in the list element area. If the list entry area exceeds 90% of its capacity, *all* log streams are offloaded to DASD. DASD offloading commences at this point, regardless of the current utilization of the log stream, and continues until an amount of data equal to the difference between the HIGHOFFLOAD threshold and the LOWOFFLOAD threshold has been offloaded.

For example, the list entry area may exceed 90% of its capacity while log stream A is only 50% utilized. Its HIGHOFFLOAD threshold is 80% and its LOWOFFLOAD threshold is 60%. Even though log stream A has not reached its HIGHOFFLOAD threshold, or even its LOWOFFLOAD threshold, data is offloaded until 20% of the log stream has been offloaded. This is the difference between 80% and 60%. After the offloading operation has completed, log stream A is at 30% utilization (50% minus 20%).

Thus, the log stream used by an application issuing very few journal write requests may be offloaded to DASD because of frequent journal write requests by other applications using other log streams in the same structure.

However, if multiple log streams share the same structure, a situation where list entry storage reaches 90% utilization should only occur where all the log streams have a similar amount of logging activity.

Frequency of DASD offloading

However, there is the likelihood of wasted list entry storage if you underestimate the value for AVGBUFSIZE in order to avoid frequent DASD offloading.

Dynamic repartitioning and the frequency of DASD offloading: Whenever a log stream connects to or disconnects from a CF structure, the structure undergoes dynamic repartitioning. This means that the space in the structure is partitioned between all the log streams connected to the structure. As more log streams connect, the less space each log stream is apportioned. This can lead to a higher frequency of DASD offloading as reduced log stream space means that the log stream HIGHOFFLOAD threshold percentages are reached more often.

Recommendations

A value of 64000 for MAXBUFSIZE should be appropriate for most environments and should be suitable for most purposes.

Limitations

If MAXBUFSIZE is set to greater than 65276, the element size is 512 bytes. With an 512-byte element, there is more likelihood of space being unused, and, therefore, wasted because of padding to the end of the last element for the log record. This likelihood lessens where records are larger and where systems are busier.

How implemented

AVGBUFSIZE and MAXBUFSIZE are parameters for use in the IXCMIAPU program which you would run to define coupling facility structures. For more information, see the *System/390 MVS Setting up a Sysplex* manual.

How monitored

The following facilities are available to monitor the data traffic to log streams on structures, and from log streams to DASD:

- The CICS log stream statistics. These provide a range of statistical information including a value for 'average bytes written per write' which you can calculate by dividing the 'TOTAL BYTES' value by the 'TOTAL WRITES' value. This may help you to tune the value for AVGBUFSIZE.
- RMF provides statistics including a value 'elements per entry' which you can calculate by dividing the 'TOTAL NUMBER OF ELEMENTS' value by the 'TOTAL NUMBER OF ENTRIES' value. This allows you to check the activity in element units on the log stream. RMF also informs you of the proportion of requests, per structure, that have been processed synchronously and asynchronously. This enables you to isolate structures that hold synchronously processed log stream requests from those that hold asynchronously processed log stream requests.
- SMF88 records. These provide a range of statistical information, including the number of bytes offloaded.

LOWOFFLOAD and HIGHOFFLOAD parameters on log stream definition

Important
This section assumes you are using log streams that use coupling facility structures. However, much of it applies also to DASD-only log streams. This is clarified in "DASD-only logging" on page 279.

Offloading to DASD data sets of data from a log stream may occur when usage of the log stream (either in the coupling facility or the staging data set) reaches its HIGHOFFLOAD limit, specified when the log stream is defined. For a system log, all records that have been marked for deletion are physically deleted; if, after this has been done, the LOWOFFLOAD limit has not been reached, the oldest active records are offloaded to DASD until LOWOFFLOAD is reached. For a general log, the oldest data is offloaded to DASD until the LOWOFFLOAD limit is reached.

There are also situations where offloading of data from the log stream data set occurs although the HIGHOFFLOAD threshold (and LOWOFFLOAD threshold in some circumstances) of the log stream has not been reached. These are:

- When the HIGHOFFLOAD threshold is reached in the staging data set. If the size of the staging data set is proportionally smaller than the log stream, the HIGHOFFLOAD threshold is reached on the staging data set before it is reached on the log stream data set.
- When the list entry area of the log stream reaches 90% of its capacity.

In these situations, the amount of data offloaded from the log stream is determined as follows:

```
(Current utilization or HIGHOFFLOAD, whichever is the greater) - LOWOFFLOAD
```

This is the percentage of the log stream data set that is offloaded.

Recommendations

Due to the different requirements that you will have for data on the system log, and data on general logs, different recommendations apply in each case.

System log

When an activity keypoint happens, CICS deletes the "tail" of the primary system log, DFHLOG. This means that data for completed units of work older than the previous activity keypoint is deleted. Data for each incomplete unit of work older than the previous activity keypoint is moved onto the secondary system log, DFHSHUNT, provided that the UOW has done no logging in the current activity keypoint interval.

To minimize the frequency of DASD offloading, try to ensure that system log data produced during the current activity keypoint interval, plus data not deleted at the previous activity keypoint, is always in the CF structure. To avoid offloading this data to DASD, you are recommended to:

- Ensure that the value of LOWOFFLOAD is greater than the space required for the sum of:
  1. The system log data generated during one complete activity keypoint interval
  2. The system log data generated (between syncpoints) by your longest-running transaction.

  and Calculate a value for LOWOFFLOAD using the following formula:

```
                trandur * 90
LOWOFFLOAD = ------------------ + 10  (where RETPD=0 is specified)
             akpintvl + trandur
```

  or

```
                trandur * 90
LOWOFFLOAD = ------------------      (where RETPD=dddd is specified)
             akpintvl + trandur
```

  where:

  – `akpintvl` is the interval between activity keypoints. It varies according to workload and its calculation should be based on peak workload activity, as follows:

```
                        AKPFREQ
akpintvl = ---------------------------------
           (N1 * R1) + (N2 * R2) + (Nn * Rn)
```

    where:

    - `N1, N2 ... Nn` is the transaction rate for each transaction (trans/sec).
    - `R1, R2 ... Rn` is the number of log records written by each transaction.

  – `trandur` is the execution time (between syncpoints) of the longest-running transaction that runs as part of the normal workload.

    If this duration is longer than `akpintvl` value, you can either:

    - Increase the value of AKPFREQ, so increasing the value of `akpintvl` (as long as this does not result in an unacceptably large CF structure size).
    - Change the application logic to cause more frequent syncpoints.

- Calculate a structure size based on a shorter transaction duration, and accept that DASD offloading occurs when the long-running transaction is used.

DFHLSCU recommends a value of 40% for DFHLOG's LOWOFFLOAD parameter. In practice, a good empirical range for this value is between 40% and 60%. Too low a value may result in physical offloading of log data from primary to secondary storage once the MVS Logger offload process has completed physical deletion of any unwanted log data during offload processing.

Conversely, too high a value may mean that subsequent offload processing occurs more frequently, as less space is freed up from primary storage during an offload operation.

If the results of the calculation from the formula given above do not lie within the range of 40% to 60%, it may be that your workload has atypical values for `trandur` or `akpintvl`.

It should be noted that log stream definition values (such as LOWOFFLOAD) should be reviewed after analysis of information such as statistics from MVS logger SMF 88 records.

- Set HIGHOFFLOAD to a maximum of 80
- Minimize the amount of log data produced between activity keypoints by specifying a low value on the AKPFREQ parameter. A value of 4000 is recommended.

General logs

The recommendations for forward recovery logs and user journals are different to those for the system log. There is no requirement here to retain logged data in the CF structure. Rather, due to the typical use of such data, you may only need a small structure and offload the data rapidly to DASD. If this is the case, allow HIGHOFFLOAD and LOWOFFLOAD to default (to 80 and 0 respectively).

How implemented

HIGHOFFLOAD and LOWOFFLOAD are parameters for use in the IXCMIAPU program which you would run to define log stream models and explicitly named individual log streams. For more information, see the *System/390 MVS Setting up a Sysplex* manual.

How monitored

SMF88 records and RMF provide a range of statistical information that helps you in the tuning of these parameters.

Staging data sets

Important

This section assumes you are using log streams that use coupling facility structures. For related information about DASD-only log streams, see "DASD-only logging" on page 279.

MVS keeps a second copy of data written to the coupling facility in a data space, for use when rebuilding a coupling facility in the event of an error. This is satisfactory as long as the coupling facility is failure-independent (in a separate CPC and non-volatile) from MVS.

Where the coupling facility is in the same CPC, or uses volatile storage, the MVS system logger supports staging data sets for copies of log stream data that would otherwise be vulnerable to failures that impact both the coupling facility and the MVS images.

Elements (groups of log records) are written to staging data sets in blocks of 4K bytes (not in 256-byte or 512-byte units as for log stream data sets).

Recommendations

Use the following formulae to help you tune the size of your staging data sets:

```
staging data set size= (NR * AVGBUFSIZE rounded up to next unit of 4096)
```

where NR is the number of records to fill the coupling facility structure. This can be calculated as follows:

```
NR = coupling facility structure size / (AVGBUFSIZE rounded up to next element)
```

Ensure that the coupling facility structure and staging data set can hold the same number of records. Staging data sets are subject to the same offloading thresholds as log streams are. It is sensible, therefore, to ensure as far as possible that offloading activity will be at the same frequency.

You are recommended to overestimate, rather than underestimate, staging data set size. To calculate staging data set size to accommodate the maximum number of records (where there is one record per element), use the following formulae:

```
maximum staging data set size = 8 * coupling facility structure size
```

where element size is 512 bytes, and

```
maximum staging data set size = 16 * coupling facility structure size
```

where element size is 256 bytes.

Investigate using DASD FastWrite facilities with a view to storing data in the DASD cache, as opposed to writing it directly to the staging data set. This also enables a faster retrieval of data should it be required. Be aware, however, that if you fill the cache, data is also then written out to the staging data set whenever data is written to the cache.

Activity keypoint frequency (AKPFREQ)

The activity keypoint frequency value (AKPFREQ) specifies the number of write operations performed to the log stream buffer before an activity keypoint is to be taken. A keypoint is a snapshot of in-flight tasks in the system at that time. During emergency restart, CICS only needs to read back for records for those tasks identified in a keypoint. CICS reads the system log backward until the first activity keypoint is encountered (which is the last activity keypoint taken).

The CICS delayed flush algorithm for log streams results in CICS writing log blocks to the system log. As activity increases on the system, the size of the log blocks increases (rather than the number of blocks written).

In summary, the AKPFREQ value determines the amount of writing to the log stream buffer between keypoint frequencies.

The default value of the AKPFREQ is 4000 log records.

Limitations

Increasing the AKPFREQ value has the following effects:

- Restart and XRF takeover times tend to increase.
- The amount of primary storage required for the system log increases.

Decreasing the AKPFREQ value has the following effects:

- Restart time may be reduced. This is particularly important in high-availability systems such as those running with XRF=YES.
- The amount of primary storage required for the system log decreases.
- Task wait time and processor cycles tend to increase.
- Paging may increase.

Although the last two effects have an impact on system performance, the impact is not overly significant.

Taking a keypoint imposes an overhead on the running system. Paging increases at keypoint time because many control blocks are scanned.

Setting the frequency to zero means that emergency restart takes longer. If AKPFREQ=0, CICS cannot perform log tail deletion until shutdown, by which time the system log will have spilled to secondary storage. As CICS needs to read the whole of the system log on an emergency restart, it needs to retrieve the spilled system log from DASD offload data sets.

AKPFREQ and MRO

In an MRO environment, the session allocation algorithm selects the lowest-numbered free session for use by the next task to run. Consequently, if a large number of sessions has been defined (perhaps to cope with peak workload requirements), the higher-numbered sessions are less likely to be used frequently during quieter periods.

In an MRO environment, CICS implements the 'implicit forget' process, an optimization of the two-phase commit. This means that when the mirror transaction at the remote end of an MRO connection completes any end-of-task processing, all information relating to the task is deleted when any new flow on that session arrives. This flow is usually the first flow for the next task or transaction allocated to run on the session as a result of the MRO session allocation algorithm.

Short term variations in the arrival rate of transactions means that some mirror transactions waiting to process an implicit forget can persist for some time. This is particularly the case where such mirror transactions have been allocated to high-numbered sessions during a peak period, now passed, of transaction arrival rate.

The keypoint program uses an appreciable amount of CPU capacity in processing persisting units of work such as those relating to mirror transactions waiting to process an implicit forget. This is exacerbated when the AKPFREQ value is low.

An optimum setting of AKPFREQ allows many of these persistent units of work to complete during normal transaction processing activity. This minimizes the CPU processing used by the keypoint program. You are therefore recommended to exercise some caution in reducing the value of AKPFREQ below the default value.

Recommendations

If you set AKPFREQ too high and thus make your keypoint frequency too low, the writing of the keypoints causes the system to slow down for only a short time. If you set AKPFREQ too low and make your keypoint frequency too high, you may get a short emergency restart time but you also incur increased processing, because more activity keypoints are processed.

You are recommended to set AKPFREQ to the default value of 4000. The optimum setting of AKPFREQ allows the whole of the system log to remain in the coupling facility.

How implemented

Activity keypoint frequency is determined by the AKPFREQ system initialization parameter. AKPFREQ can be altered with the CEMT SET SYSTEM[AKP(value)] while CICS is running.

How monitored

A message, DFHRM0205, is written to the CSMT transient data destination each time a keypoint is taken.

DASD-only logging

The primary storage used by a DASD-only log stream consists of:
- A data space owned by the MVS logger
- Staging data sets.

No data is written to coupling facility structures. In its use of staging data sets, a DASD-only log stream is similar to a CF log stream defined with DUPLEX(YES) COND(NO).

When the staging data set reaches its HIGHOFFLOAD limit, data is either deleted or offloaded until the LOWOFFLOAD limit is reached.

The following principles apply to DASD-only log streams as much as to CF log streams:

- Size system logs so that system log data produced during the current activity keypoint interval, plus data not deleted at the previous activity keypoint, is retained in primary storage
- For the system log, avoid “staging data set full” conditions and offloading to secondary storage.

The principle of sizing the staging data set for a DASD-only log stream is the same as that for sizing a staging data set for a coupling facility log stream. If you are migrating from CICS/ESA 4.1 or CICS/ESA 3.3, you are strongly recommended to use the DFHLSCU program to size your staging data sets. For information about DFHLSCU, see the *CICS Operations and Utilities Guide*.

If for any reason you cannot use DFHLSCU (perhaps you are not migrating from CICS/ESA 4.1 or CICS/ESA 3.3), use the following formula to calculate the size of

the staging data set for the system log. The formula calculates the value to be specified on the STG_SIZE parameter of the logstream definition; that is, the size is expressed as a number of 4K blocks:

```
Staging
DS size    =  (AKP duration)  *  No. of log writes per second for system log
(No. of 4K blocks)                                    CICS TS 390 AKPFREQ
where:   AKP duration =  ----------------------------
      No. of buffer puts per second
```

The values for the number of log writes per second and buffer puts per second can be taken from your CICS/ESA 4.1 statistics. (The value for log writes per second should not exceed 30.)

Chapter 24. Virtual and real storage

This chapter discusses performance tuning issues related to virtual and real storage in the following topics:

Tuning CICS virtual storage

The CICS virtual storage tuning process consists of several steps that you should take in the following order:

1. Understand the contents of the CICS address space. To determine what is good or bad in your system, you must first fully understand the contents of the CICS address space and what components of the system affect the size of each of the areas. See "Appendix F. MVS and CICS virtual storage" on page 611 for a description of the CICS address space.
2. Measure the CICS address space using one of the following tools to determine the approximate sizes of each of the areas:
   - CICS formatted dump (loader domain and storage domain only) to look at the CICS region
   - CICS storage statistics
   - The sample statistics program (DFH0STAT) to provide selected statistical information for estimating the size of your DSAs.
3. Using the results of the above measurements, determine if each of the areas of the CICS address space is within the expected sizes and select the areas that seem to be the most out of line from your expectations. Concentrate on these items; do not waste time on areas that represent only a small amount of storage improvement.
4. Evaluate the guidance given in this chapter to see whether it is applicable to your installation.

Splitting online systems: virtual storage

A method of increasing the virtual storage available to a CICS system is to split the system into two or more separate address spaces. Splitting a system can also allow you to use multiprocessor complexes to the best advantage because a system can then operate on each processor concurrently. Splitting systems can also provide

higher availability; see “Splitting online systems: availability” on page 185. See page 301 for information on using intercommunication facilities.

If data, programs, or terminals must be shared between the systems, CICS provides intercommunication facilities for this sharing. Two types of intercommunication are possible:

1. *Intersystem communication (ISC)*. ISC is implemented through the VTAM LU6.1 or LU6.2. These give program-to-program communication with System Network Architecture (SNA) protocols. ISC includes facilities for function shipping, distributed transaction processing, and transaction routing.
2. *Multiregion operation (MRO)*. MRO is implemented through MVS cross-memory facilities. An alternative method is to use operating system supervisor calls (SVCs). For communication across MVS images within a SYSPLEX, MRO/XCF is implemented using the MVS cross-system coupling facility. It includes function shipping, distributed transaction processing, and transaction routing.

The definition of too many MRO sessions can unduly increase the processor time used to test their associated ECBs. Use the CICS-produced statistics (see “ISC/IRC system and mode entries” on page 390) to determine the number of MRO sessions defined and used. For more detailed information on ISC and MRO, see the *CICS Intercommunication Guide*.

MRO also allows you to use multiprocessors more fully, and the multiple address spaces can be dispatched concurrently. MRO is implemented primarily through changes to CICS resource definitions and job control statements for the various regions. To relieve constraints on virtual storage, it may be effective to split the CICS address space in this manner.

Function shipping allows you to define data sets, transient data, temporary storage, IMS databases, or interval control functions as being remote. This facility allows applications to request data set services from a remote region (that is, the other CICS address space where the data sets are physically defined). Heavy use of VSAM and DL/I resources requires large amounts of virtual storage. If, for example, 500 VSAM KSDS data sets are removed to a remote region from the region where the application is being run, this can potentially save more than one megabyte.

The DL/I call and EXEC interfaces are supported for function shipping. CICS handles the access to remote resources and returns the requested items to a program without the need for recoding the program. Use of DL/I through DBCTL is usually a better alternative, and IMS data sharing might also be considered.

Distributed transaction processing allows direct communication between one application program and another application program, on a “send/receive” basis, much as a program communicates with a terminal. See the *CICS Distributed Transaction Programming Guide* for information about DTP.

Transaction routing allows a terminal owned by one CICS region to run a transaction that resides in another region, as if that transaction resided in the terminal-owning region.

Where useful

Most CICS systems can be split.

Limitations

Splitting a CICS region requires increased real storage, increased processor cycles, and extensive planning.

If you only want transaction routing with MRO, the processor overhead is relatively small. The figure is release- and system-dependent (for example, it depends on whether you are using cross-memory hardware), but for safety, assume a total cost somewhere in the range of 15–30KB instructions per message-pair. This is a small proportion of most transactions: commonly 10% or less.

The cost of MRO function shipping can be very much greater, because there are normally many more inter-CICS flows per transaction. It depends greatly on the disposition of resources across the separate CICS systems.

MRO can affect response time as well as processor time. There are delays in getting requests from one CICS to the next. These arise because CICS terminal control in either CICS system has to detect any request sent from the other, and then has to process it; and also because, if you have a uniprocessor, MVS has to arrange dispatching of two CICS systems and that must imply extra WAIT/DISPATCH overheads and delays.

The system initialization parameter ICVTSD (see page 207) can influence the frequency with which the terminal control program is dispatched. An ICVTSD value in the range 300–1000 milliseconds is typical in non-MRO systems, and a value in the range 150–300 is typical for MRO systems (and even lower if you are using function-shipping). Another system initialization parameter is MROLRM, which should be coded yes if you want to establish a long-running mirror task. This saves re-establishing communications with the mirror transaction if the application makes many function shipping requests in a unit of work.

You also have to ensure that you have enough MRO sessions defined between the CICS systems to take your expected traffic load. They do not cost much in storage and you certainly do not want to queue. Examine the ISC/IRC statistics to ensure that no allocates have been queued, also ensure that all sessions are being used.

Other parameters, such as MXT, may need to be adjusted when CICS systems are split. In an MRO system with function shipping, tasks of longer duration might also require further adjustment of MXT together with other parameters (for example, file string numbers, virtual storage allocation). Finally, if you plan to use MRO, you may want to consider whether it would be advantageous to share CICS code or application code using the MVS link pack area (LPA). Note that this is to save real storage, not virtual storage, and other non-CICS address spaces. Use of LPA for the eligible modules in CICS is controlled by the system initialization parameter, LPA=YES; this tells CICS to search for the modules in the LPA. For further information on the use of LPA, see “Using modules in the link pack area (LPA/ELPA)” on page 294.

Recommendations

To tune CICS to get more virtual storage, you must first tune MVS and then CICS. If, after you have tuned MVS common virtual storage, you still cannot execute CICS in a single address space, you must then consider splitting the CICS workload into multiple address spaces. Many installations find it convenient to split their CICS workload into multiple independent address spaces, where their workload is easily definable and no resource sharing is required. If it is easy to isolate application subsystems and their associated terminals, programs, and data

sets, it is reasonable to split a single CICS address space into two or more independent address spaces. They become autonomous regions with no interactions.

A system can be split by application function, by CICS function (such as a data set owning or terminal owning CICS), or by a combination of the two functions. Ideally, you should split the system completely, with no communication required between the two parts. This can reduce overheads and planning. If this is not possible, you must use one of the intercommunication facilities.

You can provide transaction routing between multiple copies of CICS. If additional virtual storage is needed, it would be reasonable, for example, to split the AOR into two or more additional CICS copies. When you have split the system either partially or completely, you can reduce the amount of virtual storage needed for each region by removing any unused resident programs. One consequence of this is reduce the size of the relevant DSA.

Admittedly, MRO uses additional processor cycles and requires more real storage for the new address spaces. Many installations have several megabytes of program storage, however, so the potential virtual storage savings are significant.

You should also remember that only a local or remote PSB can be scheduled at one time with function shipping, affecting the integrity of the combined databases. Distributed transaction processing can allow for transactions in both systems to concurrently schedule the PSBs.

MRO generally involves less overhead than ISC because the processing of the telecommunications access method is avoided. VTAM logons and logoffs can provide an alternative to transaction routing if the logons and logoffs are infrequent.

How implemented

You must define resources in the CSD (CICS system definition) data set, such as program files and terminal definitions. You must also create links to other systems, together with the connection and session definitions that substantiate such links.

MRO across the MVS sysplex

The CICS interregion communication (IRC) facility that supports MRO is enhanced to exploit the cross— system coupling facility (XCF) of MVS, to provide dynamic add of connections, and to rationalize MRO security.

The main benefit of adding XCF/MRO to the CICS interregion communication facility is to provide efficient and flexible CICS-to-CICS communications in an MVS sysplex environment. By exploiting the MVS cross-system coupling facility, CICS supports MRO links between MVS images, enabling you to use transaction routing, function shipping, and distributed program link across MRO links in a sysplex environment, replacing the need to use CICS ISC links through VTAM for these functions. XCF/MRO consumes much less CPU resources than ISC. A sysplex consists of multiple MVS systems, coupled together by hardware elements and software services. In a sysplex, MVS provides a platform of basic multisystem services that multisystem applications like CICS can exploit. As an installation's workload grows, additional MVS systems can be added to the sysplex to enable the installation to meet the needs of the increased workload.

You can also use XCF/MRO for distributed transaction processing, provided the LU6.1 protocol is adequate for your purpose.

Maximum task specification (MXT)

The MXT system initialization parameter limits the total number of concurrent user tasks in the CICS system. It also affects the amount of storage allocated to the kernel stack segment.

Effects

MXT primarily controls virtual storage usage, particularly to avoid short-on-storage (SOS) conditions. It also controls contention for resources, the length of queues (this can avoid excessive processor usage), and real storage usage.

MXT controls the number of user tasks that are eligible for dispatch. When MXT is set (either at startup, when an EXEC CICS SET SYSTEM command is processed, or when using a CEMT transaction) the kernel and dispatcher attempt to preallocate sufficient control blocks to guarantee that MXT user tasks can be created concurrently. The majority of the storage used in this preallocation is obtained from the CDSA or ECDSA, although a small amount of MVS storage is required for each task (approximately 256 bytes above the 16MB line, and 32 bytes below the 16MB line for each user task). It is interrelated with the DSA size limits that you set (DSALIM, EDSALIM).

Limitations

If you set MXT too low, throughput and response time can suffer when system resources (processor, real storage, and virtual storage) are not constrained.

If you set MXT too high at startup, CICS forces a smaller maximum number of tasks consistent with available storage.

If you set MXT too high while running, you get the error message: "CEILING REACHED".

For more information about MRO considerations, and the secondary effects of the region exit interval (ICV), see "Region exit interval (ICV)" on page 190.

Recommendations

Initially, set MXT to the number of user tasks you require concurrently in your system by totaling the following:

- The number of concurrent long-running tasks
- Each terminal running conversational tasks
- An estimate of the number of concurrent tasks from terminals running nonconversational tasks
- An estimate of the number of concurrent nonterminal tasks.

How implemented

The MXT system initialization parameter has a default value of 5, and a minimum setting of 1. It can be altered with either CEMT or EXEC CICS SET SYSTEM MAXTASKS commands while CICS is running.

How monitored

The CICS transaction manager statistics show the number of times the MXT ceiling has been reached.

Transaction class (MAXACTIVE)

Transaction classes give you a mechanism to limit the number of CICS tasks within your system. By spreading your tasks across a number of transaction classes and controlling the maximum number of tasks that can be dispatched within each transaction class, you can control resource contention between tasks and limit the number of tasks that CICS considers eligible for dispatching at task attach.

Effects

Together with MXT, transaction classes control the transaction "mix", that is, it ensures that one type of transaction does not monopolize CICS.

When the number of tasks within a class is at the specified ceiling, no additional tasks within that class are attached until one of them terminates.

Transaction classes can be used to force single-threading of a few tasks, either to avoid ENQ interlocks or because of the excessive effect of several such tasks on the rest of the system.

Limitations

Transaction classes are unsuitable in normal use for conversational transactions, because the (n+1) user may be locked out for a long time.

If TRANCLASS is specified with transaction CATD, the MAXACTIVE attribute of the transaction class must have a value of at least two in the corresponding field to prevent all the CATD transactions stacking up behind the one in the ECB wait during an emergency restart. See the *CICS Resource Definition Guide* for more details of TRANCLASS.

Recommendations

The MAXACTIVE attribute of the transaction class definition can be used to control a specific set of tasks that may be heavy resource users, tasks of lesser importance (for example, "Good morning" broadcast messages), and so on, allowing processor time or storage for other tasks.

By selecting transaction classes and their MAXACTIVE values, you can control the mix of transactions; that is, you can ensure that one type of transaction does not monopolize CICS. In particular, you can restrict the number of "heavyweight" tasks, the load on particular data sets or disk volumes, and the printer load on lines. For example, you can use transaction classes to isolate "difficult" tasks, or put all user tasks into separate classes. Suggested classes are simple enquiries, complex enquiries or short browses, long browses, short updates, long updates. Separate nonconversational tasks from conversational tasks. If you need to single-thread non-reentrant code, use ENQ for preference.

Using transaction classes can be useful for particularly high-resource-consuming tasks that do not exceed MAXACTIVE ceiling frequently, but should not be implemented for normal tasks or for design reasons such as serializing a function within a particular task. Application design should be reviewed as an alternative in these cases.

How implemented

You specify the maximum number of tasks in each transaction class using the MAXACTIVE attribute. You specify the value of the class associated with a

particular task using the CEDA transaction definition with the TRANCLASS attribute. Most CICS Cxxx transaction identifiers are not eligible.

MAXACTIVE values can be changed using the CEMT SET TRANCLASS(classname) MAXACTIVE(value) or EXEC CICS SET TRANCLASS() MAXACTIVE() commands.

How monitored

If you have divided your tasks into classes, you can use the CEMT INQUIRE TCLASS command to provide an online report. The CICS transaction class statistics show the number of times that the number of active transactions in the transaction class reached the MAXACTIVE value ("Times MaxAct").

CICS defines two Tclasses for its own use, DFHTCLSX and DFHTCLQ2. For information about the effects these have, see "Using transaction classes DFHTCLSX and DFHTCLQ2" on page 305.

Transaction class purge threshold (PURGETHRESH)

The PURGETHRESH attribute of the transaction class definition limits the number of tasks which are newly created, but cannot be started because the MAXACTIVE limit has been reached for the associated transaction class. These tasks are queued by the transaction manager domain in priority order until they obtain class membership.

They occupy small amounts of storage, but if the queue becomes very long CICS can become short-on-storage and take a considerable time to recover. Systems where a heavy transaction load is controlled by the TRANCLASS mechanism are most prone to being overwhelmed by the queue.

The tasks on the queue are not counted by the MXT mechanism. MXT limits the total number of tasks that have already been admitted to the system within TRANCLASS constraints.

Effects

The length of the queue of tasks waiting to be started in a TRANCLASS is limited by the PURGETHRESH attribute of that class. Any new transaction which would cause the limit to be reached is abended with the abend code AKCC. Tasks that were queued before the limit was reached are allowed to continue waiting until they can be executed.

Where useful

The PURGETHRESH attribute should be specified only where the transaction load in a TRANCLASS is heavy. This is the case in a system which uses a terminal-owning region (TOR) and multiple application-owning regions (AORs) and where the TRANCLASSes are associated with the AORs and are used to control the numbers of transactions attempting to use the respective AORs. In this configuration, an AOR can slow down or stall and the associated TRANCLASS fills (up to the value defined by MAXACTIVE) with tasks that are unable to complete their work in the AOR. New transactions are then queued and the queue can grow to occupy all the available storage in the CICS DSA within a few minutes, depending on the transaction volume.

Recommendations

The size of each entry in the queue is the size of a transaction (256 bytes) plus the size of the TIOA holding any terminal input to the transaction. There can be any number of queues, one for each TRANCLASS that is installed in the TOR.

You can estimate a reasonable size purge threshold for the queue by multiplying the maximum length of time you are prepared for users to wait before a transaction is started by the maximum arrival rate of transactions in the TRANCLASS.

Make sure that the queues cannot occupy excessive amounts of storage at their maximum lengths.

The PURGETHRESH queuing limit should not be set so low that CICS abends transactions unnecessarily, for example when an AOR slows down due to a variation in the load on the CPU.

How implemented

The PURGETHRESH attribute of a TRANCLASS is used to set the limit of the queue for that transaction class. The default action is not to limit the length of the queue.

Note that the CEMT SET TRANCLASS(name) PURGETHRESH(p) command can be used to change the purge threshold of a transaction class online.

How monitored

To monitor the lengths of the queues for each transaction class you should use CICS transaction class statistics. Many statistics are kept for each transaction class. Those that are particularly relevant here are:

XMCPI
Number of transactions abended AKCC because the size of the queue reached the PURGETHRESH limit.

XMCPQT
The peak number of transactions in the queue.

XMCTAPT
The number of times the size of the queue reached the PURGETHRESH limit.

You can also tell how many tasks are queued and active in a transaction class at any one time by using the CEMT INQUIRE TRANCLASS command.

You can monitor the number of AKCC abends in the CSMT log. These abends indicate the periods when the queue limit was reached. You must correlate the transaction codes in the abend messages with the transaction classes to determine which limit was being reached. The tasks on the queue are not counted by the MXT mechanism. MXT limits the total number of tasks that have already been admitted to the system within TRANCLASS constraints.

Task prioritization

Prioritization is a method of giving specific tasks preference in being dispatched.

Priority is specified by terminal in:

A CEDA TERMINAL definition (TERMPRIORITY)

A transaction in a CEDA TRANSACTION definition (PRIORITY)

A user in the priority field of the user segment of the external security manager (ESM), (OPPRTY).

The overall priority is determined by summing the priorities in all three definitions for any given task, with the maximum priority being 255.

```
TERMPRIORITY+PRIORITY+OPPRTY <= 255
```

The value of the PRTYAGE system initialization parameter also influences the dispatching order; for example, PRTYAGE=1000 causes the task's priority to increase by 1 every 1000ms it spends on the ready queue.

Effects

With CICS, the dispatching priority of a task is reassessed each time it becomes ready for dispatch, based on clock time as well as defined priority.

A task of priority n+1 that has just become ready for dispatch is usually dispatched ahead of a task of priority n, but only if PRTYAGE milliseconds have not elapsed since the latter last became ready for dispatch.

Thus, a low priority task may be overtaken by many higher priority tasks in a busy system, but eventually arrives at the top of the ready queue for a single dispatch.

The lower the value of PRTYAGE, the sooner this occurs.

Where useful

Prioritization is useful for browsing tasks, and tasks that use a lot of processor time. Input/Output bound tasks can take the required amount of CPU, and move on to the next read/write wait. CPU-intensive tasks take higher priority over the less intensive tasks.

Prioritization can be implemented in all CICS systems. It is more important in a high-activity system than in a low-activity system. With careful priority selection, an improvement in overall throughput and response time may be possible.

Prioritization can minimize resource usage of certain resource-bound transactions.

Limitations

Prioritization increases the response time for lower-priority tasks, and can distort the regulating effects of MXT and the MAXACTIVE attribute of the transaction class definition.

Priorities do not affect the order of servicing terminal input messages and, therefore, the time they wait to be attached to the transaction manager.

Because prioritization is determined in three sets of definitions (terminal, transaction, and operator), it can be a time-consuming process for you to track many transactions within a system.

CICS prioritization is not interrupt-driven as is the case with operating system prioritization, but simply determines the position on a ready queue. This means that, after a task is given control of the processor, the task does not relinquish that control until it issues a CICS command that calls the CICS dispatcher. After the dispatch of a processor-bound task, CICS can be tied up for long periods if CICS requests are infrequent. For that reason, prioritization should be implemented only if MXT and the MAXACTIVE attribute of the transaction class definition adjustments have proved to be insufficient.

Recommendations

Use prioritization sparingly, if at all, and only after you have already adjusted task levels using MXT and the MAXACTIVE attribute of the transaction class definition.

It is probably best to set all tasks to the same priority, and then prioritize some transactions either higher or lower on an exception basis, and according to the specific constraints within a system.

Do not prioritize against slow tasks unless you can accept the longer task life and greater dispatch overhead; these tasks are slow, in any case, and give up control each time they have to wait for I/O.

Use small priority values and differences. Concentrate on transaction priority. Give priority to control operator tasks rather than the person, or at least to the control operator's signon ID rather than to a specific physical terminal (the control operator may move around).

Consider for *high* priority a task that uses large resources. However, the effects of this on the overall system need careful monitoring to ensure that loading a large transaction of this type does not lock out other transactions.

Also consider for *high* priority those transactions that cause enqueues to system resources, thus locking out other transactions. As a result, these can process quickly and then release resources. Examples of these are:

- Using intrapartition transient data with logical recovery
- Updating frequently used records
- Automatic logging
- Tasks needing fast application response time, for example, data entry.

Lower priority should be considered for tasks that:

- Have long browsing activity
- Are process-intensive with minimal I/O activity
- Do not require terminal interaction, for example:
  - Auto-initiate tasks (except when you use transient data intrapartition queues that have a destination of terminal defined and a trigger level that is greater than zero).
  - Batch update controlling tasks.

PRTYAGE should usually be left to its default value, unless certain transactions get stuck behind higher priority transactions during very busy periods.

How implemented

You specify the priority of a *transaction* in the CEDA TRANSACTION definition with the PRIORITY attribute. You specify the priority for a *terminal* in the CEDA terminal definition with the TERMPRIORITY attribute. You specify the priority for an *operator* with the OPPRTY operand in the user segment of the external security manager (ESM).

PRTYAGE is a system initialization parameter.

How monitored

There is no direct measurement of transaction priority. Indirect measurement can be made from:

- Task priorities
- Observed transaction responses
- Overall processor, storage, and data set I/O usage.

Simplifying the definition of CICS dynamic storage areas

CICS allocates dynamic storage areas automatically. This removes the need to specify the size of each individual dynamic storage area. You need specify only the overall limits within which CICS can allocate storage for these areas.

CICS uses eight separate dynamic storage areas:

CDSA
RDSA
SDSA
UDSA
ECDSA
ERDSA
ESDSA
EUDSA

To facilitate continuous operations, and to simplify CICS system management, the individual DSA sizes are determined by CICS, and can be varied dynamically by CICS as the need arises. You simply specify how much storage that CICS is to use for the DSAs in two amounts—one for the four DSAs above the 16MB boundary, and the other for the four DSAs below. Automatic sizing within the specified limits removes the need for restarts to change DSA sizes. You can also vary the overall limits dynamically, using either the CEMT master terminal command, or an EXEC CICS SET command. See “The dynamic storage areas” on page 620 for more information about considerations regarding the size you should set for the DSALIM and EDSALIM parameters.

Extended dynamic storage areas

Conceptually, you should view the system initialization parameter, EDSALIM, as limiting the size of one large storage pool where each of the DSAs above the line (ECDSA, ESDSA, EUDSA, ERDSA) acquire space. The unit of allocation is 1MB extents. An allocated extent can be used by only the owning EDSA (EDSAs cannot share a given extent). If there is not enough space within the allocated extents to satisfy a request, additional extents are acquired as necessary unless the EDSA limit has been reached.

In situations where one of the EDSAs attempts to acquire an additional extent and there are no free extents, empty extents belonging to other EDSAs are used. Program compression may be triggered when EDSALIM is approached and there are few free or empty extents available. The EUDSA no longer contains programs, and so program compression does not occur in it. The other EDSAs are evaluated individually to determine if program compression is required.

Estimating EDSALIM

Specify EDSALIM so that there is sufficient space to accommodate all the EDSAs.

- The EDSAs (ECDSA, ESDSA, EUDSA and ERDSA) are managed by CICS as part of EDSALIM. Because the EDSAs are managed in 1 megabyte increments (extents), it is important to allow for fragmentation and partially used extents by rounding up the value of EDSALIM accordingly. Because there are 4 extended DSAs, consider rounding up each EDSA's requirement to a megabyte boundary.
- If TRANISO=NO, you must allow 64K per concurrent active task for the EUDSA. The safest estimate is to assume MXT as the number of concurrent active tasks. If your applications use more than 64K per task, you must adjust the formulas accordingly (use multiples of 64K increments if adjusting the formula).
- If TRANISO=YES, you must allow 1 megabyte per concurrent active task for the EUDSA. Again, the safest estimate would be to assume MXT as the number of concurrent active tasks. If your applications use more than 1 meg per task, you must adjust the formulas accordingly (use multiples of 1 meg increments if adjusting the formula).

Two methods of estimating EDSALIM are shown below. Information can be obtained by looking at your current storage manager statistics (see the DSA limit in the storage manager statistics, dynamic storage areas and in the task subpools).

Kernel stack storage is allocated out of EDSA, and for more information about kernel storage see "CICS kernel storage" on page 634.

Note: In each of the components of the calculations that follow remember to round their values up to a megabyte boundary.

1. If you would like to specify a generous EDSA limit:

   For TRANISO=NO:
   ECDSA + ERDSA + EUDSA + (64K * MXT)

   For TRANISO=YES:
   ECDSA + ERDSA + EUDSA + (1MB * MXT)

2. If your current installation EDSALIM and MXT values are set to values larger than necessary:

   For TRANISO=NO:
   Peak ECDSA Used + Peak ERDSA Used + (Peak EUDSA Used) - (EUDSA Peak Page Storage in Task Subpools) + (64K * (Peak number of tasks))

   For TRANISO=YES:
   Peak ECDSA Used + Peak ERDSA Used + (Peak EUDSA Used) - (EUDSA Peak Page Storage in Task Subpools) + (1M * (Peak number of tasks))

The minimum EDSALIM is 10MB and the default value is 20MB. The maximum EDSALIM size is (2 gigabytes - 1 megabyte).

These are guidelines for specifying initial values for the EDSA limit. The EDSALIM can be dynamically adjusted using the CEMT command without having to stop and restart your CICS system. The safest approach is to:

- Slightly over-specify EDSALIM initially.
- Monitor each EDSA's usage while your system is running near peak loads.
- Tune your EDSALIM size using CEMT SET SYSTEM commands.

If you under-specify EDSALIM, your system can go short on storage and it you may not be able to issue CEMT commands to increase the limit. If this happens you can use CPSM to increase the EDSA limit.

Dynamic storage areas (below the line)

If your installation is constrained for virtual storage below the line, the simplest approach is to set DSALIMI equivalent to the sum of the CDSA and UDSA. You will have to consider adjusting these figures so that they use the 256KB limit, see "DSA details".

You may find that there is slightly more storage available below the line for DSA storage. CICS pre-allocates approximately 3KB or less of kernel stack storage below the line per task. The majority of kernel stack storage is allocated out of CICS DSAs instead of MVS storage.

DSA details

The DSAs below the line are managed in a similar manner to the EDSAs. The differences in DSA and EDSA management are:

- The extent size for the CDSA, RDSA, and SDSA is in 256KB increments rather than the 1MB size used for the EDSAs.
- If transaction isolation is active, the extent size for the UDSA is 1MB and each UDSA extent must be aligned on a megabyte boundary. If translation isolation is not active, the allocation is in 256KB extents. It is important to keep this in mind because you must allow for some fragmentation between the 256KB extents of the CDSA, RDSA and SDSA compared with the 1 megabyte extents of the UDSA.
- Task storage is 4KB per active task in the UDSA compared with the 1 megabyte or 64KB size for the EUDSA.
- If your applications use more than 4KB per task, you must adjust the formula accordingly (use multiples of 4KB increments if adjusting the formula).
- If your system uses the SDSA and the RDSA, you must allow for these DSAs to be allocated in 256KB increments.

Estimating DSALIM

If you have sufficient virtual storage to adjust your DSA limit to a value greater than the sum of your current CDSA + UDSA, the following formulas may be used

Note: In each of the components of the calculations that follow remember to round their values up to a 256KB boundary.

1. If you can afford to specify a generous DSA limit:

   CDSA + UDSA + 256K (if both RDSA and SDSA used)

2. If your current installation DSALIM and MXT values are set to values larger than necessary:

   Peak CDSA Used + Peak UDSA Used + 256K (if both RDSA and SDSA used)

The minimum DSALIM is 2MB and the default value is 5MB. (The maximum DSALIM size is 16MB).

As discussed in the EDSALIM section, it is safer to slightly over-specify DSALIM than to under-specify it. DSALIM can be tuned to a smaller value after you have obtained data from your running system.

Dynamically altering DSALIM value

Accurate sizing of DSALIM and EDSALIM parameters is no longer critical. It is not necessary to recycle your CICS system to make a change to the DSA sizes. CEMT SET SYSTEM, EXEC CICS SET SYSTEM, or CEMT SET DSAS, a new CEMT panel which groups all the storage-related parameters together, can be used to make a change. Care should always be taken, however, when increasing DSALIM or EDSALIM, as other subsystem problems may occur. For example, an MVS getmain could fail. It is necessary to understand the storage requirement outside the DSAs.

A reduction of DSALIM or EDSALIM cannot take place if there are no DSA extents free to MVS FREEMAIN. The storage manager will MVS FREEMAIN extent as they become available until the new DSALIM or EDSALIM value is reached. A short-on-storage condition may occur when reducing DSALIM or EDSALIM. A new parameter, SOSSTATUS, has been added to CEMT INQUIRE SYSTEM, EXEC CICS INQUIRE SYSTEM, and CEMT INQUIRE DSAS, to give you some indication of short-on-storage conditions.

Using modules in the link pack area (LPA/ELPA)

Some CICS management and user modules can be moved into the link pack area (LPA) or the extended link pack area (ELPA). For systems running multiple copies of CICS, this can allow those multiple copies to share the same set of CICS management code.

Effects

The benefits of placing code in the LPA or ELPA are:

- The code is protected from possible corruption by user applications. Because the LPA or ELPA is in protected storage, it is virtually impossible to modify the contents of these programs.
- Performance can be improved and the demand for real storage reduced if you use the LPA or ELPA for program modules. If more than one copy of the same release of CICS is running in multiple address spaces of the same processor, each address space requires access to the CICS nucleus modules. These modules may either be loaded into each of the address spaces or shared in the LPA or ELPA. If they are shared in the LPA or ELPA, this can reduce the working set and therefore, the demand for real storage (paging).
- You can decrease the storage requirement in the private area by judicious allocation of the unused storage in the LPA or ELPA created by rounding to the next segment.

Limitations

Putting modules in the LPA or ELPA requires an IPL of the operating system. Maintenance requirements should also be considered. If test and production systems are sharing LPA or ELPA modules, it may be desirable to run the test system without the LPA or ELPA modules when new maintenance is being tested.

The disadvantage of placing too many modules in the LPA (but not the ELPA) is that it may become excessively large. Because the boundary between the CSA and the private area is on a segment boundary, this means that the boundary may move down one megabyte. The size of the ELPA is not usually a problem.

Recommendations

Use the SMP/E USERMOD called LPAUMOD to select those modules that you want to use for the LPA. This indicates the modules that are eligible for LPA or ELPA. You can use this USERMOD to move the modules into your LPA library.

The objective is to use the LPA wisely to derive the maximum benefit from placing modules in the LPA.

All users with multiple CICS address spaces should put all eligible modules in the ELPA.

How implemented

LPA=YES must be specified in the system initialization table (SIT). Specifying LPA=NO allows you to test a system with new versions of CICS programs (for example, a new release) before moving the code to the production system. The production system can then continue to use modules from the LPA while you are testing the new versions.

An additional control, the PRVMOD system initialization parameter, enables you to exclude particular modules explicitly from use in the LPA.

For information on installing modules in the LPA, see the *CICS Transaction Server for OS/390 Installation Guide*.

Map alignment

CICS maps that are used by basic mapping support (BMS) can be defined as aligned or unaligned. In aligned maps, the length field associated with a BMS data field in the BMS DSECT is always aligned on a halfword boundary. In unaligned maps, the length field follows on immediately from the preceding data field in the map DSECT.

A combination of aligned and unaligned maps can be used.

Effects

In unaligned maps, there is no guarantee that the length fields in the BMS DSECT are halfword-aligned. Some COBOL and PL/I compilers, in this case, generate extra code in the program, copying the contents of any such length field to, or from, a halfword-aligned work area when its contents are referenced or changed.

Specifying map alignment removes this overhead in the application program but increases the size of the BMS DSECT, at worst by one padding byte per map data field, and marginally increases the internal pathlength of BMS in processing the map. The best approach, therefore, is to use unaligned maps, except where the compiler being used would generate inefficient application program code.

In COBOL, an unaligned map generates an unsynchronized structure. In PL/I, an unaligned map generates a map DSECT definition as an unaligned structure. Correspondingly, aligned maps produce synchronized structures in COBOL and aligned structures in PL/I.

Some of the VS COBOL compilers have an option that does not generate the extra copy statements associated with an unsynchronized structure, but other COBOL compilers do. If this option is available, it should be specified because you do not then need aligned maps.

Limitations

In CICS, BMS maps are always generated in groups ("map sets"). An entire map set must be defined as aligned or unaligned. Also, maps may be used by application programs written in a variety of languages. In these cases, it is important to choose the option that best suits the combination of programs and, if there is any requirement for both aligned and unaligned maps, the ALIGNED option should be taken.

Conversion of maps from aligned to unaligned or conversely should be avoided if possible, because changing the map DSECT also requires reassembly or recompilation of all application programs that reference it.

How implemented

Map alignment is defined when maps are assembled. Aligned maps use the SYSPARM(A) option. The BMS=ALIGN/UNALIGN system initialization parameter defines which type of map is being used.

The map and map set alignment option can also be specified when maps and map sets are defined using the screen definition facility (SDF II) licensed program product. For more information, see the *Screen Definition Facility II Primer for CICS/BMS Programs*.

How monitored

The importance of map alignment may be found by inspecting programs that handle screens with a large number of fields. Try recompiling the program when the BMS DSECT is generated first without, and then with, the map alignment option. If the program size, as indicated in the linkage edit map, drops significantly in the second case, it is reasonable to assume there is high overhead for the unaligned maps, and aligned maps should be used if possible.

Resident, nonresident, and transient programs

Programs, map sets, and partition sets can be defined as RESIDENT(NO | YES) and USAGE(NORMAL | TRANSIENT). Programs can be defined as RELOAD(NO | YES).

Effects

Any program defined in the CSD is loaded into the CDSA, RDSA, SDSA, ECDSA, ERDSA, or ESDSA on first usage. RELOAD(YES) programs cannot be shared or reused. A program with RELOAD(YES) defined is only removed following an explicit EXEC CICS FREEMAIN. USAGE(TRANSIENT) programs can be shared, but are deleted when the use count falls to zero. RESIDENT(NO) programs become eligible for deletion when the use count falls to zero. The CICS loader domain progressively deletes these programs as DSA storage becomes shorter, on a least-recently-used basis.

RESIDENT(YES) programs are not normally deleted. If NEWCOPY is executed for any program, a new copy is loaded and used on the next reference and the old copy becomes eligible for deletion when its use count falls to zero.

On a CICS warm start, an initial free area for the various resident program subpools is allocated. The size of this area is based on the total lengths of all currently loaded resident programs as recorded during the preceding CICS shutdown. When a resident program is loaded, CICS attempts to fit it into the initial free area. If it does not fit, it is loaded outside the initial free area, and the space inside the initial free area remains unallocated until other (smaller) resident programs are loaded into it. This could occur if a resident program has increased its size since it was last loaded (before the last CICS shutdown). If the program in question is very large, storage problems could occur because of the large amount of unused storage in the initial free area allocated for resident programs.

Recommendations

Because programs that are not in use are deleted on a least-recently-used (LRU) basis, they should be defined as RESIDENT(NO) unless there are particular reasons to favor particular programs by keeping them permanently resident. Variations in program usage over time are automatically taken account of by the LRU algorithm.

Thus, a much-used nonresident program is likely to remain resident anyway, whereas – during periods of light usage – a resident program could be wasting the virtual storage it permanently occupies.

For programs written to run above the 16MB line, you should be able to specify EDSALIM large enough such that virtual storage is not a constraint.

If a program is very large or frequently updated such that its size increases, consider defining it as non-resident and issuing a LOAD with the HOLD option as part of PLTPI processing. The program will not be released during program compression, but also ensures that there will not be a significant amount of initial free storage reserved for resident programs which may go unused because the new (larger) program will not fit into it.

The reasons for defining a program as RESIDENT might be:

- Possible avoidance of storage fragmentation, because all such programs are in a single block of storage (but not new copies of programs).
- Programs are needed to deal with potential crises (for example, CEMT).
- Heavy contention on the DFHRPL program libraries. However, this should usually be dealt with by data set placement or other DASD tuning, or use of MVS library lookaside to maintain program copies in an MVS dataspace. See "Use of LLA (MVS library lookaside)" on page 192.

How monitored

The tuning objective is to optimize throughput at an acceptable response time by minimizing virtual storage constraint. There are specific loader domain statistics for each program.

Putting application programs above the 16MB line

CICS Transaction Server for OS/390 Release 3 keeps RMODE(ANY) application programs in the EDSA, which is in MVS extended virtual storage above the 16MB line. Work areas associated with the programs may also reside above the 16MB line.

Effects

It is possible to LINK or XCTL between 31-bit mode programs and 24-bit mode programs. You can convert programs to 31-bit mode programs and move them above the 16MB line to the extended private area. Moving programs above the 16MB line frees that amount of virtual storage below the 16MB line for other use.

See the *CICS Operations and Utilities Guide* for information on using programs from the LPA or extended link pack area (ELPA).

Using the ELPA is usually better than using the extended private area when multiple address spaces are employed, because the program is already loaded when CICS needs it, and real-storage usage is minimized.

When running a CICS system with transaction isolation enabled, performance benefits can be gained by moving transactions and application programs above the line. Program work areas are then obtained from the EUDSA with a 1MB pagesize rather than the UDSA which has a 4KB pagesize.

Where useful

This facility is useful where there is demand for virtual storage up to the 16MB line and there is sufficient real storage.

Limitations

Because the purpose of using virtual storage above the 16MB line is to make the space below this available for other purposes, there is an overall increase in the demand for real storage when programs are moved above the 16MB line.

There is a restriction on the use of COMMAREAs being passed between programs running in 31-bit addressing mode and programs running in 24-bit addressing mode. COMMAREAs passed from a 31-bit program to a 24-bit program must be capable of being processed by the 24-bit program, therefore they must not contain 31-bit addresses: addresses of areas that are themselves above the 16MB line.

How implemented

Programs that are to reside above the 16MB line must be link-edited with the AMODE(31),RMODE(ANY) options on the MODE statement of the link-edit. See the *CICS Operations and Utilities Guide* for further information.

Transaction isolation and real storage requirements

When using transaction isolation there is a cost in terms of real storage. Paging problems can result if insufficient real storage is allocated, which then affects performance. The cost is very much based on the number of subspaces in use in the system, and the size of EDSALIM.

Since the pagesize of the EUDSA is one MB, EDSALIM is likely to be very large for a CICS system which has transaction isolation active. Since this virtual storage needs to be mapped with page and segment tables using real storage, an increase in the real storage usage can occur. In addition to the real storage used to map the virtual storage for the EDSALIM, subspaces also require real storage. For example:

- Each subspace requires 2.5 pages.

- Assuming each transaction in the system requires a unique subspace, (transaction definition TASKDATAKEY(USER) and ISOLATE(YES)), real storage required is MXT * 2.5 pages.
- If each transaction in the system requires a page of storage in the EUDSA (1MB page), a page table is required to map the storage. Real storage is MXT * 1 page.
- A further three pages are required to give a total of Real storage = MXT * (1 + 2.5 pages) + 3 pages.
- All of this real storage is allocated from the ELSQA.

The figures for the real storage usage is in addition to that required for a CICS system that does not have transaction isolation active.

Note: Where a page means a 4KB page of real storage.

Limiting the expansion of subpool 229 using VTAM pacing

Subpool 229 may be expanded if batch type terminals send data faster than a CICS transaction can process that data. The use of secondary to primary pacing, sometimes called inbound pacing, limits the amount of data queued in subpool 229 for any given batch terminal.

PACING controls the flow of traffic from the network control program (NCP) to the terminal and does not affect the processor activity as such. VPACING on the other hand controls the flow of traffic between the host and the NCP.

The VPACING parameter of the CICS APPL statement determines how many messages can be sent in a session to the VTAM application program by another VTAM logical unit without requiring that an acknowledgment (called a "pacing response") be returned. The host sends data path information units (PIUs) according to the definition of VPACING. The first PIU in a group carries a pacing indicator in the RH. When this PIU is processed by the NCP, the NCP sends a response to the host with the same pacing indicator set to request a new pacing group. This means that, for every *x* PIUs to a terminal and every *y* PIUs to a printer, the pacing response traffic must flow from the NCP to the host which, based on the volume of traffic, could cause a significant increase in host activity.

Normally, VPACING is implemented when a shortage of NCP buffers requires controlling the volume of flow between the host and the NCP. You may be able to lessen the effect on the processor by increasing the VPACING value to what the NCP can actually tolerate.

The PACING parameter is required for most printers, to match the buffer capacity with the speed of printing the received data. Terminals do not normally require pacing unless there is a requirement to limit huge amounts of data to one LU, as is the case with some graphics applications. Use of pacing to terminals causes response time degradation. The combination of PACING and VPACING causes both response time degradation and increased processor activity, and increased network traffic.

Recommendations

PACING and VPACING should be specified for all terminals to prevent a "runaway" transaction from flooding the VTAM network with messages and requiring large amounts of buffer storage. If a transaction loops while issuing SENDs to a terminal, IOBUF (CSA storage) and NCP buffers may fill up causing slowdowns and CSA shortage conditions.

PACING and VPACING should always be specified high enough so that normal data traffic may flow without being regulated, but excessive amounts of data are prevented from entering the network and impairing normal data flow.

How implemented

For secondary to primary pacing, you must code:

- SSNDPAC=nonzero value in the LOGMODE entry pointed to by the secondary application program
- VPACING=nonzero value on the APPL definition for the secondary application.

The value used is coded on the VPACING parameter. If either of these values are zero, no pacing occurs.

Specify VPACING on the APPL statement defining the CICS region, and any nonzero value for the SSNDPAC parameter on the LU statement defining the batch device. You should ensure that the device supports this form of pacing by referring to the component description manual for that device.

For further information on the selection criteria for values for the PACING and VPACING parameters, see the *ACF/VTAM Version 2 Planning and Installation Reference* manual.

Chapter 25. MRO and ISC

This chapter discusses performance tuning issues related to multiregion operation, and ISC.

CICS intercommunication facilities

CICS intercommunication facilities allow different CICS systems to communicate and share resources with each other. These facilities consist of the following components:

- Function shipping
- Distributed transaction processing
- Asynchronous processing
- Transaction routing.
- Distributed program link.

For details of the CICS intercommunication facilities, see the *CICS Intercommunication Guide*. See also "Splitting online systems: virtual storage" on page 281, and "Splitting online systems: availability" on page 185.

If there are a number of intercommunication requests for each transaction, function shipping generally incurs the most overhead. The number of requests per transaction that constitutes the break-even point depends on the nature of the requests.

Both distributed transaction processing (DTP) and asynchronous processing are, in many cases, the most efficient method of intercommunication because a variety of requests can be batched in one exchange. DTP, however, requires an application program specifically designed to use this facility. For information about designing and developing DTP, see the *CICS Distributed Transaction Programming Guide*.

Transaction routing, in most cases, involves one input and one output between systems, and the overhead is minimal.

Multiregion operation (MRO), in general, causes less processor overhead than intersystem communication (ISC) because the SVC pathlength is shorter than that through the multisystem networking facilities of VTAM. This is particularly true with CICS MRO, which provides a long-running mirror transaction and fastpath transformer program.

Some SVC-processing overhead can be eliminated from MRO in CICS with the use of MVS cross-memory services. Cross-memory services use the MVS common system area (CSA) storage for control blocks, not for data transfer. This can also be of benefit. Note, however, that MVS requires that an address space using cross-memory services be nonswappable.

For situations where ISC is used across MVS images, consider using XCF/MRO. XCF/MRO consumes less processor overhead than ISC.

Your sysplex configuration may offer you a choice of XCF connectivity for XCF/MRO interregion communication. You should check whether your sysplex allows a choice of channel-to-channel connectivity, or whether you can use coupling facility channel links only. If you have a choice, channel-to-channel links generally offer better performance for XCF/MRO operations

ISC mirror transactions can be prioritized. The CSMI transaction is for data set requests, CSM1 is for communication with IMS/ESA systems, CSM2 is for interval control, CSM3 is for transient data and temporary storage, and CSM5 is for IMS/ESA DB requests. If one of these functions is particularly important, it can be prioritized above the rest. This prioritization is not effective with MRO because any attached mirror transaction services any MRO request while it is attached.

If ISC facilities tend to flood a system, this can be controlled with the VTAM VPACING facility. Specifying multiple sessions (VTAM parallel sessions) increases throughput by allowing multiple paths between the systems.

CICS also allows you to specify a VTAM class of service (COS) table with LU6.2 sessions, which can prioritize ISC traffic in a network. Compare the performance of CICS function shipping with that of IMS/ESA data sharing.

Limitations

- Use of intercommunication entails trade-offs as described in “Splitting online systems: virtual storage” on page 281 and “Splitting online systems: availability” on page 185.
- Increased numbers of sessions can minimally increase real and virtual storage but reduce task life. The probable overall effect is to save storage.
- MVS cross-memory services reduce CSA and cycle requirements.
- MRO high performance facilities reduce processing requirements.
- IMS/ESA data sharing usually reduces processor requirements.
- Accessing DL/I databases via the IMS DBCTL facility reduces processor requirements relative to function shipping.
- For MRO considerations, read about the secondary effects of the region exit interval (ICV) on page “Region exit interval (ICV)” on page 190.

How implemented

See the *CICS Transaction Server for OS/390 Installation Guide* for information about resetting the system for MRO or ISC. See also “Splitting online systems: virtual storage” on page 281.

How monitored

CICS ISC/IRC statistics (see page 390) show the frequency of use of intercommunication sessions and mirror transactions. The VTAM trace, an SVC trace, and RMF give additional information.

Intersystems session queue management

When intersystems links are added to the system there is the possibility that they cannot respond adequately to transaction requests because the remote system is performing badly. The poor performance can be due either to a long-term condition such as lack of resource or overloading, or a temporary situation such as a dump being taken. In any case there is the danger that the problem can cause a long queue to form in the requesting system.

Mechanisms are provided in CICS for:

- Protection of the requesting system from using too many resources whilst transactions queue for the use of the intersystems sessions.
- Detection of problems in remote systems. CICS can issue messages to indicate a problem on an intersystems connection and the parameters control the criteria that are used to determine when a problem exists, or has gone away.

The two mechanisms are:

1. The QUEUELIMIT and MAXQTIME parameters on the connection resource definition.

   The QUEUELIMIT parameter limits the number of transactions which can be queued in allocate processing waiting for a session to become free. Any transactions which try to join a queue already at its limit are rejected.

   The MAXQTIME parameter is a control on the wait time of queued allocate requests that are waiting for free sessions on a connection that appears to be unresponsive. If the rate of processing of the queue indicates that a new allocate will take longer than the specified time to reach the head of the queue, the whole queue is purged.
2. The XZIQUE user exit, which is given control when an allocate request is about to be queued, or the first time it succeeds after a suspected problem. The XZIQUE exit can control the queue in the same way as the CEDA parameters, or you can use it to add more sophisticated controls of your own.

Both mechanisms produce the same effect on the application program which issued the allocate; a SYSIDERR condition is returned. Return codes are also provided to the dynamic routing program to indicate the state of the queue of allocate requests.

The *CICS Resource Definition Guide* contains more description of the CEDA commands; and the *CICS Customization Guide* gives programming information about the XZIQUE exit and its relationship with the rest of CICS, including application programs and the dynamic routing program.

Relevant statistics

For each connection CICS records the following:

- The number of allocates queued for the connection, and the peak value of this number. (`Peak outstanding allocates` in the Connection statistics.)

  You can use this statistic to see how much queuing normally takes place on connections in your system. If there is occasionally a large queue you should consider controlling it. "Are enough sessions defined?" on page 55 has more advice on setting the right number of sessions for your connections.

For each of the queue control mechanisms CICS records the following statistics for each connection:

- The number of allocates which were rejected due to the queue becoming too large
- The number of times the queue was purged because the throughput was too slow
- The number of allocates purged due to slow throughput.

"ISC/IRC system and mode entry statistics" on page 52 also contains an explanation of these, and other connection statistics.

Ways of approaching the problem and recommendations

The queue limit mechanism should be used to control the number of tasks waiting for the use of an intersystems link. You should use the control to ensure that even at its maximum length the queue does not use too many, and certainly not all, of the MXT slots in the system. You can also use the MAXACTIVE setting of a TRANCLASS definition to do this if you can segregate your transactions into classes that correspond to the remote regions they require.

You should allow sufficient intersystems sessions to enable their free availability during normal running. Session definitions do not occupy excessive storage, and the occupancy of transaction storage probably outweighs the extra storage for the session. The number of sessions should correspond to the peak number of transactions in the system which are likely to use the connection—you can see the maximum number of sessions being used from the terminal statistics for the connection. If all sessions were used, the connections statistics show the number of times allocates were queued compared with the total number of requests.

Even in a system that has no problems, there are significant variations in the numbers of transactions that are active at any time, and the actual peak number may be larger than the average over a few minutes at the peak time for your system. You should use the average rather than the actual peak; the queueing mechanism is intended to cope with short-term variations, and the existence of a queue for a short time is not a cause for concern.

The start of a queue is used by the queue limiting mechanism as a signal to start monitoring the response rate of the connection. If queues never form until there is a big problem, the detection mechanism is insensitive. If there are always queues in the system, it will be prone to false diagnosis.

You should set the queue limit to a number that is roughly the same size as the number of sessions—within the limits imposed by MXT if there are many connections whose cumulative queue capacity would reach MXT. In this latter case, you might need to design your own method—using ZXIQUE—of controlling queue lengths so that the allocation of queue slots to connections is more dynamic.

You should set the MAXQTIME parameter with regard to the time you think the users of the system should be prepared to wait for a response in the case of a potential problem, but bear in mind that you should not set it to be short in combination with a queue limit that is low, because this leads to a very sensitive detection criterion.

Monitoring the settings

The number of allocates rejected by the queue control mechanism should be monitored. If there are too many, it may indicate a lack of resources to satisfy the demands on the system—or poor tuning.

The number of times the queue is purged should indicate the number of times a serious problem occurred on the remote system. If the purges do not happen when the remote system fails to respond, examine the setting of the MAXQTIME parameter—it may be too high, and insensitive. If the indication of a problem is too frequent and causes false alarms simply due to variations in response time of the remote system, the parameter may be too low, or the QUEUELIMIT value too low.

Using transaction classes DFHTCLSX and DFHTCLQ2

DFHTCLSX and DFHTCLQ2 in RDO group DFHISCT allow you to control the amount of storage used by CICS to execute the CLS1/2 and CLQ2 transactions respectively.

Effects

These tasks execute the activities needed to acquire an APPC conversation (CLS1/2), and to resynchronize units of work for MRO and APPC connections (CLQ2). Usually there are not many tasks, and they need no control. However, if your CICS system has many connection definitions, these may be acquired simultaneously as a result of initializing the system at startup, or as a result of a SET VTAM OPEN, or SET IRC OPEN command.

How implemented

The system definitions are optional. Install resource group DFHISCT to activate them. As supplied, the MAXACTIVE parameter in the DFHTCLSX and DFHTCLQ2 is 25. This should give sufficient control to prevent the system reaching a short-on-storage situation. (Tasks CLS1 and CLS2 each require 12K of dynamic storage, and CLQ2 tasks require up to 17K). The purge threshold should not be set to a non-zero number, and the maxactive should not be set to 0. They both prevent CICS executing tasks necessary to intersystems functions.

It is not advisable to set the MAXACTIVE value too low because network delays or errors may cause one of the tasks in the TCLASS to wait and block the use of the TCLASS by succeeding transactions. Setting a low value can also extend shutdown time in a system with a large number of connections.

Terminal input/output area (SESSIONS IOAREALEN) for MRO sessions

For MRO function shipping, the SESSIONS definition attribute, IOAREALEN, is used. This attribute regulates the length of the terminal input/output area (TIOA) to be used for processing messages transmitted on the MRO link. These TIOAs are located above the 16MB line.

Effects

The IOAREALEN value controls the length of the TIOA which is used to build a message transmitted to the other CICS system (that is, an outgoing message).

Two values (value1 and value2) can be specified. Value1 specifies the initial size of the TIOA to be used in each session defined for the MRO connection. If the size of the message exceeds value1, CICS acquires a larger TIOA to accommodate the message.

Only one value is required, however if value2 is specified CICS will use value2 whenever the message cannot be accommodated by the value1.

A value of zero causes CICS to get a storage area exactly the size of the outgoing message, plus 24 bytes for CICS requirements.

If the IOAREALEN value is not specified, it defaults to 4KB.

Where useful

The IOAREALEN attribute can be used in the definition of sessions for either MRO transaction routing or function shipping. In the case of MRO transaction routing, the value determines the initial size of the TIOA, whereas the value presents some tuning opportunities in the MRO function shipping environment.

Limitations

Real and virtual storage can be wasted if the IOAREALEN value is too large for most messages transmitted on your MRO link. If IOAREALEN is smaller than most messages, or zero, excessive FREEMAIN and GETMAIN requests can occur, resulting in additional processor requirements.

Recommendations

For optimum storage and processor utilization, IOAREALEN should be made slightly larger than the length of the most commonly encountered formatted application data transmitted across the MRO link for which the sessions are defined. For efficient operating system paging, add 24 bytes for CICS requirements and round the total up to a multiple of 64 bytes. A multiple of 64 bytes (or less) minus 24 bytes for CICS requirements ensures a good use of operating system pages.

How implemented

The TIOA size can be specified in the IOAREALEN attribute of the SESSIONS definition.

Batching requests (MROBTCH)

Certain events in a region can be accumulated in a batch prior to posting, until the number specified in the MROBTCH system initialization parameter is reached (or ICV times out). Then, the region is started so that it can process the requests. The batching of MRO requests includes some non-MRO events such as:

- VSAM physical I/O completion
- Subtasked (mostly VSAM) request completion (if SUBTSKS=1 is specified)
- DL/I request completion implemented through DBCTL.

Strictly speaking, batching is applicable to a TCB rather than the region. MROBTCH is applied only to the 'quasi-reentrant' mode TCB.

Effects

Compared to no batching (MROBTCH=1, that is, the default), setting MROBTCH=n has the following effects:

- Up to [(n–1)*100/n]% saving in the processor usage for waiting and posting of that TCB. Thus, for n=2, 50% savings may be achieved, for n=3, 66% savings, for n=6, 83% savings, and so on.
- An average cost of (n+1)/2 times the average arrival time for each request actually batched.

- Increased response time may cause an increase in overall virtual storage usage as the average number of concurrent transactions increases.
- In heavily loaded systems at peak usage, some batching can happen as a natural consequence of queueing for a busy resource. Using a low MROBTCH value greater than one may then decrease any difference between peak and off-peak response times.

Setting MROBTCH higher than 6 is not recommended as the decreasing additional processor saving is unlikely to be worth the further increased response time.

You require a relatively low value of MROBTCH for ICV to maintain reasonable response time during periods of low utilization.

Recommendations

Depending on the amount of response time degradation you can afford, you can set MROBTCH to different values using either CEMT or EXEC CICS SET SYSTEM MROBATCH(value).

The recommendation is to use CEMT or EXEC CICS INQUIRE SYSTEM MROBATCH(value) to arrive at a suitable batch value for a given workload. See the *CICS Supplied Transactions* manual for more information about CEMT; for programming information about the EXEC CICS system programming commands, see the *CICS System Programming Reference* manual.

During slow periods the ICV unconditionally dispatches the region, even if the batch is not complete and provides a minimum delay. In this case, set ICV to 500 milliseconds in each region.

Extending the life of mirror transactions (MROLRM)

This MROLRM system initialization parameter can have a significant effect on the performance of a workload in an MRO function shipping environment.

Setting **MROLRM=NO** causes the mirror to be attached and detached for each function-shipped request until the first request for a recoverable resource or a file control start browse is received. After such a request is received, the mirror remains attached to the session until the calling transaction reaches syncpoint.

Setting **MROLRM=YES** in a region receiving function shipping requests causes a mirror transaction to remain attached to the MRO session from first request until the calling transaction reaches syncpoint. This option causes system-dependent effects, as follows:

- Some systems show significant improvements in processor utilization per transaction. They are likely to be systems with a significant percentage of inquiry transactions, each with multiple VSAM calls, or transactions with many reads followed by a few updates.
- Some systems show no performance difference. Workloads using IMS/ESA, or transactions that make a lot of use of VSAM-update or browse-activity, may fall into this category.
- Some systems could be degraded because there is an extra flow at syncpoint. An example of this would be a system with a very simple inquiry transaction workload.

In general, setting MROLRM=YES is recommended.

Deletion of shipped terminal definitions (DSHIPINT and DSHIPIDL)

In a transaction routing environment, terminal definitions can be ″shipped″ from a terminal-owning region (TOR) to an application-owning region (AOR). A shipped terminal definition in an AOR becomes redundant when:

- The terminal user logs off.
- The terminal user stops using transactions which route to the AOR.
- The TOR on which the user is signed on is shut down.
- The TOR is restarted without recovering autoinstalled terminal definitions, and the autoinstall user program DFHZATDX assigns a new set of terminal ids to the same set of terminals.

Shipped terminal definitions which have become redundant may need to be deleted. Long-lasting shipped terminal definitions do not generally cause storage problems because of the relatively small amounts of storage which they occupy. However, there are other considerations, such as security, which may require that redundant shipped terminal definitions are not allowed to persist in an AOR.

The CICS-supplied transaction CRMF periodically scans the shipped terminal definitions in the AOR and flags those which it has determined to be redundant. If any redundant definitions have been identified, the CICS-supplied transaction CRMD is invoked to delete them. This processing is referred to as the CICS timeout delete mechanism.

The system initialization parameters DSHIPINT and DSHIPIDL control the amount of time for which a redundant shipped terminal definition is allowed to survive and the frequency at which shipped terminal definitions are tested for redundancy.

Effects

The DSHIPIDL system initialization parameter determines the period of time for which a shipped terminal definition is allowed to remain inactive before it may be flagged for deletion. The DSHIPINT system initialization parameter determines the time interval between invocations of the CRMF transaction. CRMF examines all shipped terminal definitions to determine which of them have been idle for longer than the time interval specified by DSHIPIDL. If CRMF identifies any redundant terminal definitions, it invokes CRMD to delete them.

Where useful

The CRMF/CRMD processing is most effective in a transaction routing environment in which there may be shipped terminal definitions in an AOR which remain idle for considerable lengths of time.

Limitations

After CRMF/CRMD processing has deleted a shipped terminal definition, the terminal definition must be re-shipped when the terminal user next routes a transaction from the TOR to the AOR. Take care, therefore, not to set DSHIPIDL to a value that is low enough to cause shipped terminal definitions to be frequently deleted between transactions. Such processing could incur CPU processing costs, not just for the deletion of the shipped terminal definition, but also for the subsequent re-installation when the next transaction is routed.

Consider that a large value chosen for DSHIPINT, influences the length of time that a shipped terminal definition survives. The period of time for which a shipped terminal definition remains idle before deletion is extended by an average of half

of the DSHIPINT value. This occurs because a terminal, after it has exceeded the limit for idle terminals set by the DSHIPIDL parameter, has to wait (for half of the DSHIPINT interval) before CRMF is scheduled to identify the terminal definition as idle and flag it for CRMD to delete. When the DSHIPINT interval is significantly longer than the DSHIPIDL interval (which is the case if the default values of 120000 for DSHIPINT and 020000 for DSHIPIDL are accepted), DSHIPINT becomes the dominant factor in determining how long an idle shipped terminal definition survives before being deleted.

Recommendations

Do not assign too low a value to DSHIPIDL. The storage occupied by the shipped terminal definitions is not normally a concern, so the default value, which specifies a maximum idle time of 2 hours is reasonable, unless other concerns (such as security) suggest that it should be shorter.

Decide whether you wish to delete idle shipped terminal definitions incrementally or altogether. CRMF processing in itself causes negligible CPU overhead, so a low value for DSHIPINT may therefore be specified at little cost, if a sensible value for DSHIPIDL has been chosen. Specifying a low value for DSHIPINT so that CRMF runs relatively frequently could mean that idle terminal definitions are identified in smaller batches, so that CRMD processing required to delete them is spread out over time.

A higher value for DSHIPINT, especially if the default value of 12 hours is accepted, may mean that CRMF identifies a considerable number of idle terminal definitions, so that a larger burst of CPU is required for the CRMD processing. To ensure that this type of processing occurs during periods of low activity in the CICS region, the CEMT INQUIRE/SET/PERFORM DELETSHIPPED commands (and their equivalent SPI commands) are available to help you schedule when the CRMF transaction will be invoked.

How implemented

The maximum length of time for which a shipped terminal definition may remain idle before it can be flagged for deletion is specified by the CICS system initialization parameter DSHIPIDL. The interval between scans to test for idle definitions is specified by the CICS system initialization parameter DSHIPINT.

Both these parameters can be adjusted by the CEMT INQUIRE/SET DELETSHIPPED commands. Note that the revised interval to the next invocation of the timeout delete mechanism starts from the time the command is issued, not from the time it was last invoked, nor from the time of CICS startup.

The timeout delete mechanism may be invoked immediately by the CEMT PERFORM DELETSHIPPED command or its SPI equivalent.

How monitored

The CICS terminal autoinstall statistics provide information on the current setting of the DSHIPINT and DSHIPIDL parameters, the number of shipped terminal definitions built and deleted, and the idle time of the shipped terminal definitions.

Chapter 26. Programming considerations

This chapter discusses performance tuning issues related to programming in the following sections:

BMS map suffixing and the device-dependent suffix option

CICS BMS allows you to use different versions of a map set for different device types by specifying a one-letter suffix for each different device type. Use of this facility requires the BMS device-dependent suffix (DDS) option. For further information on map set suffixes, see the *CICS Application Programming Guide*.

Where only one version of a map is involved, it is optional whether the device type suffix is coded. If the DDS option is being used, it is more efficient to use the device suffixes than to leave the suffix blank. This is because, if the DDS option applies, CICS first looks for a map set with a suffix name and then searches again for a map with a blank suffix. Processor cycle requirements are reduced by eliminating the second table lookup.

Effects

If only one device type is used with all maps in a CICS system and all devices have the same screen size, CICS can be initialized to look for a blank suffix, thus eliminating the second lookup.

If the map is to be used with multiple devices, multiple maps with the same basic source are needed because the device type needs to be specified, and suffixing is required in this case.

Recommendation

If you decide that you need device-dependent suffixing, you should suffix all your map sets. If you do not need it, use blank suffixes (no suffix at all) and specify the NODDS option in BMS.

How implemented

Maps are named in the link-edit process. These names are defined in the MAPSET definition. Specifying NODDS in the BMS= system initialization parameter determines that map suffixing is not used in CICS.

How monitored

No direct measurement of map suffixing is given.

COBOL RESIDENT option

Compiling online COBOL programs with the optional RESIDENT and mandatory NODYNAM compiler options (available with the VS COBOL Version 4 and VS COBOL II compilers) allows those application programs to share common COBOL library subroutines. The library subroutines are not compiled into each module; instead, linkage is set up to a common set of subroutines and so only one copy of each module exists in a system rather than one copy for each program.

Effects

The CICS translator automatically inserts a CBL card with the RESIDENT option. NODYNAM must be specified.

Under the RESIDENT option, any library routines required by the COBOL program are not link-edited. Instead, COBOL initialization code tries to locate them when the program (or subprogram) is first invoked, by issuing MVS LOAD macros. If the library routines have been set up in the link pack area (LPA), MVS simply passes the routine addresses to COBOL, which then inserts them into the COBOL program, without any extra I/O. But if the routines are not in the LPA, they have to be loaded from the disk libraries (STEPLIB, JOBLIB, and LINKLIB).

The COBOL execution-time routines for programs compiled with RES,NODYNAM are loaded with an MVS LOAD macro (SVC) only if they are not already loaded.

When a COBOL program is initialized, it unconditionally loads ILBCBL00, which is loaded only the first time it is referenced. This module has a vector table of module names and addresses. When a module is needed, a search is made to check whether that module is already loaded. If it has been loaded previously, its address is used. If the module that is referenced has not been loaded, it is loaded by MVS and its address is placed in the vector table for future reuse.

These modules are loaded from the LINKLIB, any LINKLST data set, JOBLIB, STEPLIB, or perhaps from the LPA, but ***not*** from DFHRPL (COBOL does not know anything about DFHRPL). The MVS LOAD is not issued with any DCB parameter, so only the standard MVS LOAD hierarchy can be used. They should not be defined on the CSD, because this is not known to COBOL either. RES,NODYNAM is a COBOL feature, ***not*** a CICS feature.

In the CICS environment, many of the required COBOL library routines can be placed in the LPA; this gives an environment similar to that of the PL/I shared library. This reduces the size of each COBOL program according to its subroutine requirement. The saving in space normally offsets any extra space needed for one copy of each library subroutine in the LPA (assuming the modules have not already been put in the LPA for use by batch COBOL programs).

ILBCBL00 must ***not*** be in the LPA, however. It is loaded into the user's region and has a set of vector addresses for the other modules that may be used by the COBOL programs within that region. When a new module is requested, its address is not known and the COBOL interface routine loads that module and places its address in the list to be used again, as explained above.

If the COBOL RESIDENT option is used under CICS and the desired COBOL subroutines are not LPA-resident, an MVS LOAD is issued for each such subroutine when the first COBOL CICS program to require that subroutine is initialized. This causes a synchronous halt to CICS until the I/O is complete.

Using the RESIDENT option saves real and virtual storage. Approximately 3.5KB of storage can be saved per program, depending on the release of COBOL used and what subroutines are referenced. These subroutines are loaded the first time they are referenced, or can reside in the LPA and be shared by all programs that reference them.

Note that the requirement is that the ILBxxxxx routines be reentrant, not the application code. This feature is purely a COBOL feature and CICS does not have any specific code to support it.

The ability of CICS to share code between multiple concurrent users is based on pseudoreentrance. This means that the program must be reentrant at the time you pass control to CICS with a command but not between times.

Limitations

Recompilation of programs is required for programs not using these options.

Recommendations

Ensure that the resident subroutines are placed in the link pack area so that an MVS LOAD is not incurred the first time they are referenced.

How implemented

Resident subroutines are implemented by specifying RESIDENT and NODYNAM when the program is compiled.

How monitored

A link-edit map shows storage savings. RMF shows overall real and virtual storage usage.

PL/I shared library

The PL/I optimizing compiler has a facility whereby those resident library modules likely to be used in more than one program simultaneously can be stored together in the link pack area, from where they can be invoked from any region. This facility, known as the *PL/I shared library*, is available to PL/I programs running as CICS applications, provided that they were compiled by the PL/I optimizing compiler. The PL/I shared library is another facility that helps the user to conserve storage.

PL/I resident library routines can be shared between multiple CICS PL/I programs, rather than being compiled into each separate PL/I application program. This can save real and virtual storage; the amount depending on the number of resident library routines that each program uses.

If you want to use these routines but your programs are not compiled to share routines, you must recompile them and all programs must use the same level of the PL/I compiler. Programs compiled to use this facility do so automatically if the shared library is specified.

How implemented

To run PL/I application programs with the PL/I shared library, ensure that you generate the PL/I shared library modules. CICS looks for the presence of the significant shared library interface routines at startup time.

How monitored

A link-edit map shows storage savings. RMF shows overall real and virtual storage usage.

VS COBOL II

VS COBOL II programs can be loaded above the 16MB line and can also use most working storage above the 16MB line.

VS COBOL II has library modules that are grouped together in COBPACKs that need to be defined to CICS. COBPACKs can be tailored by the installation.

How implemented

One item of tailoring recommended is to have COBPACKs IGZCPPC and IGZCPAC contain only AMODE(31), RMODE(ANY) modules so that they may be loaded above the 16MB line.

VS COBOL II also has a tuning mechanism in IGZTUNE which specifies the initial amount of storage to be GETMAINed below (and above) the 16MB line.

This mechanism should be used, defining an optimum value that does not:

- Waste space below the 16MB line
- Incur unnecessary additional GETMAINs because the initial amount was too small.

How monitored

Use of IGZOPT allows VS COBOL II to report on the effect of IGZTUNE options. Production systems should ensure that such reporting is turned off.

Language Environment

Language Environment -conforming CICS applications issuing EXEC CICS LINK requests cause an increase in system pathlength. Repeated EXEC CICS LINK calls to the same Language Environment-conforming program result in multiple GETMAIN/FREEMAIN requests for run-time unit work areas (RUWAs).

RUWAPOOL(YES) results in the creation of a run-unit work area pool during task initialization. This pool is used to allocate RUWAs required by Language Emvironment-conforming programs. This reduces the number of GETMAINS and FREEMAINS in tasks which perform many EXEC CICS LINKS to Language Environment-conforming programs.

For more information, about the RUWAPOOL system initialization parameter, see the *CICS System Definition Guide*.

If Language Environment/370 is active in an address, the runtime libraries of the native language, such as COBOL, PL/I, are not needed. This means that CICS has a single interface to all the language run times.

Language Environment run time options for AMODE (24) programs

The default Language Environment run time options for CICS are ALL31(ON) and STACK(ANY). This means that all programs must run above the line (AMODE(31))

in a Language Environment. To allow AMODE(24) programs to run in an LE environment, ALL31(OFF) and STACK(BELOW) can be specified. However, if you globally change these options so that all programs can use them, a lot of storage will be put below the line, which can cause a short-on-storage condition.

Using DLLs in C++

When each dynamic link library (DLL) is first loaded, the cost of initialization can be determined by the size of writable static area required by the DLL. Initialization costs can be reduced by removing unnecessary items from the writable static area.

When using DLLs you should consider the following:

- Specifying the #pragma variable (x,NORENT). This places some read —only variables such as tables in the code area.
- Specifying #pragma strings(readonly). This works for C code whose default is that literal strings are modifiable. C++ already has literal strings as read only by default.
- Examine the prelinker map to see what the big areas are. If you find, for example, @STATICC, you have unnamed writable static objects such as strings or static variables.

CICS automatic dynamic storage tuning

APAR PQ39052
Documentation changed 20/10/2000

You can optionally, through the AUTODST system initialization parameter, activate dynamic storage tuning. When this function is active, CICS requests Language Environment to accumulate the actual storage requirements of each main program execution. At the end of each program execution, Language Environment returns to CICS the program's storage allocations, which are then used to pre-allocate storage for subsequent executions of the program. Note that this mechanism operates on storage allocations, not on the actual storage used by the user application program, as in the case of the Language Environment CEECSTX exit. However, although it operates on storage allocations rather than the actual storage used by an application program, it incurs less overhead than the tuning method based on the CEECSTX exit.

For more information, see the.*Language Environment for OS/390 and VM Programming Guide*.

Chapter 27. CICS facilities

This chapter discusses performance tuning issues related to the various CICS facilities as follows:

CICS temporary storage (TS)

CICS temporary storage is a scratchpad facility that is heavily used in many systems. Data in temporary storage tends to be short-lived, emphasis being placed on ease of storage and retrieval. Temporary storage exists in two forms:

- Main temporary storage, which is in the dynamic storage area above the 16MB line (ECDSA)
- Auxiliary temporary storage is stored in a VSAM-managed data set while the storage for the buffers is allocated from the ECDSA.

Temporary storage is used in many circumstances within CICS, as well as for requests from application tasks. The uses of temporary storage include:

- Basic mapping support (BMS) paging
- Message switching (CMSG transaction) or BMS routing
- Interval control: EXEC CICS START FROM (...) to hold data until it is retrieved
- Execution diagnostic facility (EDF) to review prior pages of diagnostic information
- MRO/ISC local queueing while the target system is unavailable
- Your applications for:
  - Scratchpad
  - Queueing facility
  - Data transfer.
- Other products or application packages.

Effects

If main temporary storage is used, requests to a TS queue are serialized with the storage being allocated from the ECDSA.

The performance of auxiliary temporary storage is affected by the characteristics of the data set where it resides. The VSAM control interval (CI) size affects transfer efficiency, with a smaller size being desirable if access to CIs is random, and a larger size if use of CIs is more sequential. In general, the larger the queues and write/read ratio, the more sequential the usage tends to be. Records which span control intervals are possible. Up to 32767 buffers and 255 strings can be specified,

and overlap processing can be achieved, although a specific queue is still processed serially. The maximum control interval (CI) size is 64KB.

Temporary storage VSAM requests can be subtasked if SUBTSKS=1 is specified in the SIT. See "Subtasking: VSAM (SUBTSKS=1)" on page 237.

Auxiliary temporary storage queues can be made recoverable by defining a recoverable TSMODEL. Main temporary storage can never be recoverable.

Limitations

Increasing the use of main temporary storage, using a larger CI size, or increasing the number of buffers, increases the virtual storage needs of the ECDSA and real storage needs.

If you use auxiliary temporary storage, a smaller CI size can reduce the real storage requirements.

Recommendations

Main temporary storage

Temporary storage items are stored in the ECDSA above the 16MB line. No recovery is available. Queues are locked for the duration of the TS request.

The fact that temporary storage items are stored in main storage also means that there is no associated I/O, so we recommend main temporary storage for short-duration tasks with small amounts of data.

Auxiliary temporary storage

Auxiliary temporary storage occupies less address space than main temporary storage, and should be used for large amounts of temporary storage data, or for data that is to be held for long periods.

Temporary storage I/O occurs only when a record is not in the buffer, or when a new buffer is required, or if dictated by recovery requirements.

Secondary extents for temporary storage

On a cold start of temporary storage when the data set is empty, the data set is formatted to the end of the primary extent. Any secondary extents are not formatted. On a cold start of temporary storage when the data set is not empty or when temporary storage is not cold started, no formatting of the data set takes place.

The use of secondary extents allows more efficient use of DASD space. You can define a temporary storage data set with a primary extent large enough for normal activity, and with secondary extents for exceptional circumstances, such as unexpected peaks in activity.

It follows that you can reduce or eliminate the channel and arm contention that is likely to occur because of heavy use of temporary storage data.

Multiple buffers

The use of multiple VSAM buffers allows multiple VSAM control intervals to be available in storage at the same time. This makes it possible for the CICS temporary storage programs to service several requests concurrently, using different buffers.

The use of multiple buffers also increases the likelihood that the control interval required by a particular request is already available in a buffer. This can lead to a significant reduction in the number of input/output requests (VSAM requests) that have to be performed. (However, VSAM requests are always executed whenever their use is dictated by recovery requirements.) Note that although the use of a large number of buffers may greatly improve performance for non-recoverable TS queues, the associated buffers still have to be flushed sequentially at CICS shutdown, and that might take a long time.

The number of buffers that CICS allocates for temporary storage is specified by the system initialization parameter, TS.

The benefits of multiple buffers depend on the way an installation's auxiliary temporary storage is used. In most cases, the default TS specification in the SIT (three buffers) should be sufficient. Where the usage of temporary storage is high or where the lifetime of temporary storage data items is long, it may be worthwhile to experiment with larger numbers of buffers. The buffer statistics in the CICS temporary storage statistics give sufficient information to help you determine a suitable allocation.

In general, you should aim to minimize the number of times that a task has to wait either because no space in buffers is available to hold the required data or because no string is available to accomplish the required I/O. The trade-off here is between improvement of temporary storage performance and increased storage requirements. Specifying a large number of buffers may decrease temporary storage I/O but lead to inefficient usage of real storage and increased paging.

Concurrent input/output operations (multiple strings)

Temporary storage programs issue VSAM requests whenever real input/output is required between the buffers and the VSAM temporary storage data sets. The use of multiple VSAM strings enables multiple VSAM requests to be executed concurrently, which, in turn, leads to faster servicing of the buffers.

VSAM requests are queued whenever the number of concurrent requests exceeds the number of available strings. Constraints caused by this can thus be relieved by increasing the number of available strings, up to a maximum equal to the number of buffers.

The number of VSAM strings that CICS allocates for temporary storage is specified by the system initialization parameter, TS.

Multiple strings allow more I/O operations to be performed concurrently. Several I/O requests can be outstanding at any time, up to the number of strings specified. Allowing the number of strings to default to the number of buffers ensures that no tasks are waiting for a string. Not all strings may be used in this case, however, and this causes inefficient use of storage. You should adjust the number of strings by using the peak number in use given in the statistics.

If the device containing the temporary storage data set is heavily used, the TS system initialization parameter can be used to regulate the activity, but this leads to an increase in internal CICS waits.

Control interval (CI) sizes

You should first consider whether the control interval (CI) size for the data set is suitable for your overall system requirements.

BMS paging may be on a large-screen device. Check whether it exceeds your temporary storage CI size.

Because temporary storage can use records larger than the control interval size, the size of the control intervals is not a major concern, but there is a performance overhead in using temporary storage records that are larger than the CI size.

The parameter, CONTROLINTERVALSIZE, of the VSAM CLUSTER definition is specified when you allocate your data sets.

The control interval size should be large enough to hold at least one (rounded up) temporary storage record, including 64 bytes of VSAM control information for control interval sizes less than, or equal to, 16 384, or 128 bytes of control information for larger control interval sizes. For further information about the effect of the control interval size for CICS temporary storage, see the *CICS System Definition Guide*.

How implemented

Temporary storage items can be stored either in main storage or in auxiliary storage on DASD. Main-only support can be forced by specifying TS=(,0) (zero temporary storage buffers) in the SIT.

A choice of MAIN or AUXILIARY is available for the application programmer in the WRITEQ TS command for each queue. See the *CICS Application Programming Reference* manual for programming information about the WRITEQ command.

How monitored

The CICS temporary storage statistics show records used in main and auxiliary temporary storage. These statistics also give buffer and string information and data on I/O activity. RMF or the VSAM catalog gives additional information on data set performance.

If recovery is used for auxiliary temporary storage, PREFIX (called QUEUE name by the application programmer) is enqueued for DELETEQ TS and WRITEQ TS requests but not READQ TS. In a high-activity system, PREFIX should be monitored to ensure that a given PREFIX identifier is not a resource that is constraining your transaction throughput.

You should monitor the following:

TS buffer size
This is determined by the CI size.

TS PREFIX (QUEUE) identifiers
You should minimize their number and their duration in the system.

TS space
Make the data set allocation large enough to avoid task suspension.

Note: If the NOSPACE condition is not handled, the task is suspended until temporary storage becomes available. If the NOSPACE condition is handled (through the use of the HANDLE CONDITION NOSPACE command or the use of RESP on the WRITEQ TS command, or the WRITEQ TS NOSUSPEND command, the user receives control when the condition occurs, and can then decide whether to end the transaction normally, abend, or wait.

Number of TS buffers
This is controlled by the second parameter of the TS system initialization parameter.

Number of TS strings
This is controlled by the third parameter of the TS system initialization parameter.

The 75 percent rule

Temporary storage requests on a cold-started system (that is with no existing auxiliary data) are allocated from the start of DFHTEMP (the temporary storage dataset used for storing auxiliary data). They are processed by DFHTSP, the temporary storage program. The first control interval within DFHTEMP is used until a WRITEQ is issued that is too long to fit into the remaining space. DFHTSP then switches to use control interval two, etc. This process continues until 75% of the control intervals in DFHTEMP have data written to them.

WRITEQ requests after this point are directed back to the start of the dataset. DFHTSP maintains a bytemap representing the free space available within each control interval in the dataset at any time. DFHTSP now starts interrogating the bytemap to find a control interval that can accommodate new data at, or near to, the start of DFHTEMP. The reasoning behind this is that by now, queues written earlier in the CICS run could have been deleted. Such deleted data remains in control intervals but is no longer required. If the bytemap shows a control interval contains enough space, DFHTSP reads it into a temporary storage buffer, compresses it to move all valid records to the start of the control interval, and uses the remaining contiguous space to store the data from the new request.

Temporary storage attempts to reserve 25% of the control intervals in DFHTEMP to facilitate spanned record processing. This refers to data that is larger than the control intervals. Such requests generate 'special header' records used to represent the spanned data, and these records are the size of a whole control interval. As such, they require an empty control interval when being written.

If DFHTEMP contained fragmented data in each control interval, a special header record could not be stored. This is why DFHTSP tries to maintain a percentage of free control intervals for use by large items such as special header records.

Temporary storage data sharing

Shared temporary storage queues are stored in named pools in an MVS coupling facility. Each pool corresponds to a list structure in a coupling facility. Access to queues stored in the coupling facility is quicker than function shipping to a QOR.

A temporary storage server provides better availability than a QOR because you can have more than one temporary storage server for each pool (typically one server in each MVS image in the sysplex). If one temporary storage server or MVS image fails, transactions can be dynamically routed to another AOR on a different MVS image.

Local TS queues offer less performance overhead than a QOR. However, local queues can cause intertransaction affinities, forcing affected transactions to run in the same AOR so that they can access the local queue. This affects performance by inhibiting dynamic routing and preventing workload balancing across the AORs in the sysplex. Intertransaction affinities can be managed by a workload management function provided by CICSPlex SM, but you must provide intertransaction

affinities definitions for the affected transactions. The *CICS/ESA 3.3 XRF Guide* gives guidance about determining where the affinities are in your application programs. Temporary storage data sharing removes the need for the time and effort that this systems management demands by avoiding intertransaction affinity. In general, the overall workload balancing benefits provided by being able to use dynamic transaction routing to any AOR should outweigh any overhead incurred by the temporary storage servers.

CICS transient data (TD)

Transient data is used in many circumstances within CICS, including:

- Servicing requests made by user tasks, for example, a request to build a queue of data for later processing.
- Servicing requests from CICS, primarily to write messages to system queues for printing. Transient data should, therefore, be set up at your installation to capture these CICS messages.
- Managing the DASD space holding the intrapartition data.
- Initiating tasks based on queue trigger level specification and on records written to an intrapartition destination.
- Requesting logging for recovery as specified in your CICS transient data definitions.
- Passing extrapartition requests to the operating system access method for processing.

Various options can affect the performance of this facility.

Recovery options

Recovery can affect the length of time for which a transient data record is enqueued. You can specify one of three options:

1. *No recovery*. If you specify no recovery, there is no logging, no enqueuing for protecting resources.
2. *Physical recovery*. Specify physical recovery when you need to restore the intrapartition queue to the status that it had immediately before a system failure. The main performance consideration is that there is no deferred transient data processing, which means that automatic task initiation may occur instantaneously. Records that have been written may be read by another task immediately. CIs are released as soon as they have been exhausted. For every WRITEQ TD request, the CI buffer is written to the VSAM data set.

   Note: All other resources offering recovery within CICS provide only logical recovery. Using backout in an abend situation would exclude your physically recoverable and non-recoverable transient data from the backout.
3. *Logical recovery*. Specify logical recovery when you want to restore the queues to the status that they had before execution of the failing task (when the system failed or when the task ended abnormally). Thus, logical recovery works in the same way as recovery defined for other recoverable resources such as file control, and temporary storage.

In summary, physical recovery ensures that records are restored in the case of a system failure, while logical recovery also ensures integrity of records in the case of a task failure, and ties up the applicable transient data records for the length of a task that enqueues on them.

Up to 32767 buffers and 255 strings can be specified for a transient data set, with serial processing only through a destination.

Specifying a higher trigger level on a destination causes a smaller number of tasks to be initiated from that destination. Transient data can participate in file subtasking if SUBTSKS=1 is specified in the SIT (see "Subtasking: VSAM (SUBTSKS=1)" on page 237).

Intrapartition transient data considerations

Multiple VSAM buffers

When you use multiple buffers and strings for intrapartition transient data support, this can remove the possible constraint in transient data caused by the use of a single system-wide buffer (and string). Statistics allow you to tune the system with regard to transient data usage.

If requests have to be queued, they are queued serially by transient data destination. Typically, a request has to be queued if the control interval it requires is in use, or if one or more previous requests for the same queue or destination are already waiting. Under these conditions, the servicing of requests for other queues or destinations can continue.

The use of multiple buffers also increases the likelihood that the control interval required by a particular request is already available in a buffer. This can lead to a significant reduction in the number of real input/output requests (VSAM requests) that have to be performed. (However, VSAM requests are always executed whenever their use is dictated by the requirements of physical and logical recovery.)

The number of buffers that CICS allocates for transient data is specified by the TD system initialization parameter. The default is three.

The provision of multiple buffers allows CICS to retain copies (or potential copies) of several VSAM CIs in storage. Several transient data requests to different queues can then be serviced concurrently using different buffers. Requests are serialized by queue name, not globally. Multiple buffers also allow the number of VSAM requests to the transient data data set to be reduced by increasing the likelihood that the CI required is already in storage and making it less likely that a buffer must be flushed to accommodate new data. VSAM requests are still issued when required by recovery considerations.

The benefits of multiple buffers depend on the pattern and extent of usage of intrapartition transient data in an installation. For most installations, the default specification (three buffers) should be sufficient. Where the usage of transient data is extensive, it is worthwhile to experiment with larger numbers of buffers. The buffer statistics give sufficient information to help determination of a suitable allocation. In general, the aim of the tuning should be to minimize the number of times a task must wait because no buffers are available to hold the required data.

In this exercise, there is a trade-off between improving transient data performance and increased storage requirements. Specifying a large number of buffers may decrease transient data I/O and improve concurrency but lead to inefficient usage of real storage. Also, if there is a large number of buffers and a small number of queues, internal buffer searches per queue may take longer.

The buffers are obtained from the ECDSA during initialization.

Multiple VSAM strings

As far as concurrent input/output operations with CICS are concerned, the transient data programs issue VSAM requests whenever real input/output is required between the buffers and the VSAM transient data data sets. The use of multiple VSAM strings enables multiple VSAM requests to be executed concurrently, which in turn leads to faster servicing of the buffers.

VSAM requests are queued whenever the number of concurrent requests exceeds the number of available strings. Constraints caused by this be relieved by increasing the number of available strings, up to a maximum of 255. The limit of 255 on the number of strings should be taken into consideration when choosing the number of buffers. If the number of buffers is more than the number of strings, the potential for string waits increases.

The number of VSAM strings that CICS allocates for transient data is specified by the TD system initialization parameter. The CICS default is three.

Logical recovery

Logging and enqueuing occur with logical recovery transactions (including dynamic backout of the failing task's activity on the transient data queue). Logical recovery would generally be used when a group of records have to be processed together for any reason, or when other recoverable resources are to be processed in the same task.

During processing of the transient data request, the destination queue entry is enqueued from the first request, for either input or output, or both (if the queue is to be deleted), until the end of the UOW. This means that none of the other tasks can access the queue for the same purpose during that period of time, thus maintaining the integrity of the queue's status.

At the end of the UOW (syncpoint or task completion), syncpoint processing takes place and the queue entry is logged. Any purge requests are processed (during the UOW, a purge only marks the queue ready for purging). The empty CIs are released for general transient data use. Any trigger levels reached during the UOW cause automatic task initiation to take place for those queues that have a trigger level greater than zero. The buffer is written out to the VSAM data set as necessary.

The DEQueue on the queue entry occurs, releasing the queue for either input or output processing by other tasks. Records written by a task can then be read by another task.

Logging activity

With *physical* recovery, the queue entry is logged after each READQ, WRITEQ, and DELETEQ, and at an activity keypoint time (including the warm keypoint).

With *logical* recovery, the queue entry is logged at syncpoint and at activity keypoint time (including the warm keypoint).

Secondary extents for intrapartition transient data

During initialization of intrapartition transient data, CICS initializes a VSAM empty intrapartition data set by formatting control intervals until the first extent of the data set is filled. Additional control intervals are formatted as required if the data set has been defined with multiple extents.

The use of secondary extents allows more efficient use of DASD space. You can define an intrapartition data set with primary extents large enough for normal activity, and with secondary extents for exceptional circumstances, such as unexpected peaks in activity.

It follows that you can reduce or eliminate the channel and arm contention that is likely to occur because of heavy use of intrapartition transient data.

Extrapartition transient data considerations

Extrapartition destinations are, in practice, sequential data sets to which CICS uses QSAM PUT LOCATE or PUT MOVE commands. The main performance factor to note is the possibility of operating system waits; that is, the complete CICS region waits for the I/O completion. The wait (of long duration) can occur for one of the following reasons:

- No buffer space available.
- Secondary space allocation.
- Volume (extent) switching.
- Dynamic open or close of the data set.
- A force end of volume caused by the application.
- The data set is defined on a physical printer (1403 or 3211) and the printer has run out of paper.
- A RESERVE has been issued for another data set on the same volume.

Therefore, you should try to eliminate or minimize the occurrences of CICS region waits by:

- Having sufficient buffering and blocking of the output data set
- Avoiding volume switching by initially allocating sufficient space
- Avoiding dynamic OPEN/CLOSE during peak periods.

An alternative method of implementing sequential data sets is to employ a CICS user journal. Table 13 summarizes the differences between these two methods.

Table 13. Extrapartition transient data versus user journal

Extrapartition TD	**User Journal**
Region (CICS) may wait	Task waits
Buffer location: In MVS storage	Buffer location: In DSA
Number of buffers: 1—32767	Two buffers
Input ***or*** output	***Both*** input ***and*** output, but tasks may wait
Accessible by multiple tasks	• Accessible for output by multiple tasks • Accessible for input by single task under exclusive control

Indirect destinations

To avoid specifying extrapartition data sets for the CICS-required entries (such as CSMT and CSSL) in CSD definitions for TDQUEUES, you are recommended to use indirect destinations for combining the output of several destinations to a single destination. This saves storage space and internal management overheads.

Long indirect chains can, however, cause significant paging to occur.

Limitations

Application requirements may dictate a lower trigger level, or physical or logical recovery, but these facilities increase processor requirements. Real and virtual storage requirements may be increased, particularly if several buffers are specified.

How implemented

Transient data performance is affected by the TRIGGERLEVEL and RECOVSTATUS operands in the transient data resource definitions that have been installed.

Recommendations

Suggestions for reducing WAITS during QSAM processing are to:

- Avoid specifying a physical printer.
- Use single extent data sets whenever possible to eliminate WAITS resulting from the end of extent processing.
- Avoid placing data sets on volumes subject to frequent or long duration RESERVE activity.
- Avoid placing many heavily-used data sets on the same volume.
- Choose BUFNO and BLKSIZE such that the rate at which CICS writes or reads data is less than the rate at which data can be transferred to or from the volume, for example, avoid BUFNO=1 for unblocked records whenever possible.
- Choose an efficient BLKSIZE for the device employed such that at least 3 blocks can be accommodated on each track.

How monitored

The CICS statistics show transient data performance. CICS transient data statistics can be used to determine the number of records written or read. Application knowledge is required to determine the way in which the lengths of variable length records are distributed. RMF or the VSAM catalog shows data set performance.

Global ENQ/DEQ

Global ENQ/DEQ extends the CICS/ESA application programming interface to provide an enqueue mechanism that serializes access to a named resource across a specified set of CICS regions contained within a sysplex. Because Global ENQ/DEQ eliminates the most significant remaining cause of inter-transaction affinity, it enables better exploitation of parallel sysplex. It also reduces the need to provide intertransaction affinity rules to dynamic routing mechanisms such as CICSPlex/SM, thus reducing the system management cost of exploiting parallel sysplex.

How implemented

Global ENQ/DEQ uses OS/390 global resource serialization (GRS) services to achieve locking that is unique across multiple MVS images in a sysplex. GRS can be configured as either GRS=STAR or GRS=RING.

Recommendations

When GRS is initialized as a star configuration, all the information about resource serialization is held in the ISGLOCK coupling facility structure. GRS accesses the coupling facility when a requestor issues an ENQ or DEQ on a global names resource.

GRS=RING, however, should be used with extreme caution, as this configuration can result in serious performance constraints.

The performance impact can be for a many reasons, but primarily it is due to the delay in having the request complete the ring. The larger the number of MVS images in the ring combined with a large value for RESMIL will cause delays in the request completing the ring. The ENQ request cannot be granted until the request returns to the originating MVS image. The value specified for RESMIL (in the GRSCNF member of *SYS1.Parmlib)*should be no greater than 1, preferably 0. For performance reasons, in a sysplex of greater than 2 MVS images, a GRS STAR configuration should be used.

CICS monitoring facility

The CICS monitoring facility collects data about the performance of all user- and CICS-supplied transactions during online processing for later offline analysis. The records produced by CICS monitoring are of the MVS System Management type 110, and are written to an SMF data set.

Monitoring data is useful for performance, tuning, and for charging your users for the resources they use. See "Chapter 6. The CICS monitoring facility" on page 61 for further information.

Limitations

Performance class monitoring can be a significant overhead. The overhead is likely to be about 5 to 10%, but is dependent on the workload.

Recommendations

If you do not need accounting information because other billing processes exist and you have other means of gathering any performance data required, the CICS monitoring facility should not be used. The same is true for the exception component.

Recording of the above information incurs overhead, but, to tune a system, both performance and exception information may be required. If this is not a daily process, the CICS monitoring facility may not need to be run all the time. When tuning, it is necessary to run the CICS monitoring facility during peak volume times because this is when performance problems occur.

Consider excluding fields from monitoring records if overuse of the SMF data set is a potential problem.

How implemented

To implement CICS monitoring, you can reset the system initialization table parameters (MNPER, MNEXC, and MN)—see the *CICS System Definition Guide*.

You can change the settings dynamically using either CEMT INQUIRE|SET MONITOR or EXEC CICS INQUIRE|SET MONITOR. See "Controlling CICS monitoring" on page 67 for more information. Alternatively see the *CICS Supplied Transactions* manual for details of CEMT, and the *CICS System Programming Reference* manual for programming information about INQUIRE and SET commands.

For further information about using the CICS monitoring facility, see "Chapter 6. The CICS monitoring facility" on page 61.

How monitored

CICS Monitoring Domain statistics show the number of records produced of each type. These statistics monitor CMF activity.

MVS address space or RMF data can be gathered whether or not the CICS monitoring facility is active to give an indication of the performance overhead incurred when using the CICS monitoring facility.

CICS trace

CICS trace is used to record requests made by application programs to CICS for various services. Because this involves the recording of these requests each time they occur, the overhead depends on the frequency of the requests.

The CICS internal trace table resides in MVS virtual storage above the 16MB line (but not in the EDSAs).

A trace table always exists and is used for recording exception conditions useful for any first failure data capture. Other levels of trace are under the control of the user. There are a large number of parameters and the CEMT commands which allow dynamic control over the system and transaction dumps.

Effects

Buffers for the CICS auxiliary trace data set are allocated dynamically from MVS free storage below the 16MB line. Auxiliary trace is activated when the system initialization parameter AUXTR, or a startup override, is set on.

Buffer allocation may also take place at execution time in response to a CETR or CEMT transaction request to set auxiliary trace to START (CEMT SET AUXTRACE START) or simply to open the auxiliary trace data set. For more information, see the CEMT SET AUXTRACE section in *CICS Supplied Transactions* manual.

Deallocation or freeing of the buffer space occurs in response to CEMT SET AUXTRACE STOP command. Note that the buffer space is ***not*** freed on STOP and SWITCH requests, the former not implying CLOSE and the latter having been optimized.

Limitations

Running trace increases processing requirements considerably. Not running trace, however, reduces the amount of problem determination information that is available.

The additional cost of auxiliary trace is mainly due to the I/O operations. Auxiliary trace entries vary in size, and they are written out in blocks of 4KB. Twin buffers are used but, even if the I/O can be overlapped, the I/O rate is quite large for a busy system.

When you use CICS auxiliary trace, you may need to decrease the relevant DSALIM system initialization parameter by 8KB to ensure that adequate address space is given up to the operating system to allow for the allocation of the two 4KB auxiliary trace buffers.

Recommendations

The trace table should be large enough to contain the entries needed for debugging purposes.

With first failure data capture, CICS produces some trace entries regardless of the settings produced. Because of this most of the tracing overhead can be reduced by running with the following options:

- Internal tracing off
- Auxiliary tracing on
- Print auxiliary trace data only when required.

CICS allows tracing on a transaction basis rather than a system basis, so the trace table requirements can be reduced.

How implemented

Trace activation is specified with the INTTR system initialization parameter or as a startup override.

The size of the trace table is specified by the TRTABSZ system initialization parameter or as a startup override. The minimum size is 16KB.

Trace can be defined at the transaction level with the TRACE attribute on in the TRANSACTION definition.

Auxiliary trace activation is specified with the AUXTR system initialization parameter.

With CICS initialized and running, internal trace and auxiliary trace can be turned on or off, independently and in either order, with one of the following: CETR, CEMT SET INTRACE START or CEMT SET AUXTRACE START commands. Auxiliary trace entries are recorded only when internal trace is active.

How monitored

No direct measurement of trace is given. RMF can show processing and storage requirements.

CICS recovery

Some types of recoverable resources, when they are accessed for update, cause logging. Do not define more resources as recoverable than you need for application programming requirements, because the extra logging incurs extra I/O and processor overheads. If the resource in question does not require recovery, these overheads are unproductive.

Limitations

Specifying recovery increases processor time, real and virtual storage, and I/O requirements. It also increases task waits arising from enqueues on recoverable resources and system log I/O, and increases restart time.

Recommendation

Do not specify recovery if you do not need it. If the overhead is acceptable, logging can be useful for auditing, or if a data set has to be rebuilt.

For information on specific recoverable resources, see “CICS temporary storage (TS)” on page 317, and “CICS transient data (TD)” on page 322.

How implemented

See the *CICS Recovery and Restart Guide* for information on each resource to be specified as recoverable.

How monitored

CICS auxiliary trace shows task wait time due to enqueues. RMF shows overall processor usage. CICS monitoring data shows task wait time due to journaling.

CICS security

CICS provides an interface for an external security manager (ESM), such as RACF, for three types of security: transaction, resource, and command security.

Effects

Transaction security verifies an operator’s authorization to run a transaction. Resource security limits access to data sets, transactions, transient data destinations, programs, temporary storage records, and journals. Command security is used to limit access to specific commands and applies to special system programming commands. For example, EXEC CICS INQUIRE, SET, PERFORM, DISCARD, and COLLECT. Transactions that are defined with CMDSEC=YES must have an associated user.

Limitations

Protecting transactions, resources, or commands unnecessarily both increases processor cycles, and real and virtual storage requirements.

Recommendations

Because transaction security is enforced by CICS, it is suggested that the use of both resource security and command security should be kept to the minimum. The assumption is that, if operators have access to a particular transaction they therefore have access to the appropriate resources.

How implemented

Resource security is defined with the RESSEC(YES) attribute in the TRANSACTION definition.

Command security is defined with the CMDSEC(YES) attribute in the TRANSACTION definition.

How monitored

No direct measurement of the overhead of CICS security is given. RMF shows overall processor usage.

CICS storage protection facilities

There are three facilities available that are related to storage protection:

- Storage protect
- Transaction isolation
- Command protection.

Each offers protection as follows:

Storage protect

Protects CICS code and control blocks from being accidentally overwritten by user applications.

Transaction isolation

Offers protection against transaction data being accidentally overwritten by other user transactions.

Command protection

Ensures that an application program does not pass storage to CICS using the EXEC CICS interface, which requires updating by CICS, although the application itself cannot update the storage.

Recommendation

Storage protection, transaction isolation, and command protection protect storage from user application code. They add no benefit to a region where no user code is executed; that is, a pure TOR or a pure FOR (where no DPL requests are function-shipped).

Transaction isolation and applications

When using transaction isolation, it is necessary to "activate" pages of storage to the task's allocated subspace. Before the storage is activated to the subspace it is fetch protected and, so, the task cannot access the storage. After it is activated to the subspace allocated to the task, the task has read/write access to the storage. CICS needs to activate user storage to a subspace every time the user task getmains a new page of user key task lifetime storage. Some performance cost is involved when activating storage to a subspace, so the activity should be kept to a minimum.

Storage below the 16MB line is activated in multiples of 4KB. Storage above the line is activated in multiples of 1MB. A user task rarely requires more than 1MB of storage. So a user task that executes completely above the line only requires one activate.

It is recommended that all programs should be link edited using RMODE(ANY) and defined DATALOCATION(ANY). All transactions should be defined TASKDATALOC(ANY), thus reducing the number of storage activations. Where it is necessary to obtain storage below the line, performance can be improved by obtaining all the storage in one getmain rather than several smaller getmains. This also keeps the number of storage activates to the minimum.

Certain programming languages, such as PL/I and VS COBOL, obtain storage below the line. COBOL II offers better performance because it obtains all its storage above the line.

CICS business transaction services

Business transaction services (BTS) introduces a business transaction model to CICS.

Effects

With additions to the CICS API and the provision of new system services, BTS can be used to create a new type of program that controls the flow of many separate CICS transactions so that these individual transactions become a single business transaction.

Recommendations

Since a BTS transaction may comprise many separate CICS transactions and may also span a considerable execution time, there are no specific performance recommendations for BTS transactions. There are however, some useful general observations.

How implemented

To support BTS function, CICS keeps data in new types of data sets the local request queue (DFHLRQ) and BTS repository. The local request queue data set is used to store pending BTS requests. Each CICS region has its own data set. It is a recoverable VSAM KSDS and should be tuned for best performance like a VSAM KSDS. You may have one or more BTS repositories. A BTS repository is a VSAM KSDS and is used to hold state data for processes, activities, containers, events and timers. A BTS repository is associated with a process through the PROCESSTYPE definition. If a BTS process executes on more than one CICS region, then the BTS repository needs to be shared between those regions. It would need to be a VSAM RLS file. It too should be tuned for best performance as a VSAM RLS file.

To support the execution of the BTS processes, CICS runs one or many transactions. A BTS transaction comprises a process which itself consists of one or many activities. As each activity is run, so a CICS transaction is executed. If an activity becomes dormant, waiting for an event for example, the activity restarts after that event occurs, and a new CICS transaction is started. even if this is a continuation of the business transaction. You may see many executions of the transaction specified in a process definition in the CICS statistics for a single BTS transaction. The number of transaction executed and number and type of file accesses to the BTS repository depend on how you have chosen to use BTS services. Examination of CICS statistics reports will give you this information for your applications. You should be aware that containers are stored on the BTS repository. You need to ensure that the repository is large enough to contain all the active BTS data. This is probably best done by scaling it based on a test system.

Monitor data, DFHCBTS, can be used to collect information on activities within processes. For information about this data, see “Performance data in group DFHCBTS” on page 78

Chapter 28. Improving CICS startup and normal shutdown time

This chapter provides information about areas to check when you aim to reduce the amount of time for CICS startup and normal shutdown.

The discussion covers the following topics:

Startup procedures to be checked

Because various configurations are possible with CICS, different areas of the startup may require attention, as follows:

1. Start by defining your GCD, LCD, CSD, temporary storage data sets, or transient data intrapartition data sets, as shown in the *CICS System Definition Guide*.
2. When defining your terminals, pay attention to the position of group names within the GRPLIST. If the group containing the TYPETERMs is last, all the storage used for building the terminal definitions is held until the TYPETERMs are known and this could cause your system to go short-on-storage.

   Groups in the GRPLIST in the SIT are processed sequentially. Place the groups containing the model TERMINAL definitions followed by their TYPETERMs in the GRPLIST before the user transactions and programs. This minimizes the virtual storage tied up while processing the installation of the terminals.

   Note: All terminals are installed, even surrogate TCT entries for MRO.

   You must ensure that the DFHVTAM group precedes any TERMINAL or TYPETERM definition in your GRPLIST. It is contained in the DFHLIST GRPLIST, so adding DFHLIST first to your GRPLIST ensures this. If you do not do this, the programs used to build the TCT are loaded for each terminal, thus slowing initial and cold starts.
3. You should not have more than about 100 entries in any group defined in the CSD. This may cause unnecessary overhead during processing, as well as making maintenance of the group more difficult.
4. Make sure that changing the START= parameter does not change the default for any facilities that your users do not want to have AUTO-started. Any facility that you may want to override may be specifically coded in the PARM= on the EXEC statement, or all of them may be overridden by specifying START=(...,ALL).
5. If you do not intend to make use of the CICS Web Interface, you should make sure that WEB=NO is specified in the SIT. If WEB=YES is specified, the Web domain is activated, and there is an extra read from the CICS catalog during the setup of the CICS Web Interface.
6. If you do not intend to make use of the CICS Web Interface or the Secure Sockets Layer, you should make sure that TCPIP=NO is specified in the SIT. If TCPIP=YES is specified, the Sockets domain task control block is activated.

7. Tune the VSAM parameters of the local and global catalogs to suit your installation.
   a. CI sizes should be changed for optimum data and DASD sizes (see "Size of control intervals" on page 225 for more information). 2KB index CI, and 8KB or 16KB data CI can be recommended; 32KB data has been found to slow down the COLD start.
   b. We recommend that you specify the BUFNI and BUFND parameters in your JCL for the GCD via the AMP= parameter, rather than using BUFSPACE.
   c. Alter the number of index buffers by coding the number of strings plus the number of index set records in the index. The number of records in the index set can be calculated from IDCAMS LISTCAT information as follows:
   - T = total number of index records (index REC-TOTAL)
   - D = data control interval size (data CISIZE)
   - C = data control intervals per control area (data CI/CA)
   - H = data high-used relative byte address (data HURBA)
   - The number of index set records can then be computed:

     The number of sequence set records: S = H / (D X C)
   - This calculation is really the number of used control areas. The number of sequence set records must be the same as the number of used CAs.

     The number of index set records: I = T - S

   Free space has no effect, so do not spend time trying to tune this.
8. On cold and initial starts, CICS normally has to delete all the resource definition records from the global catalog. You can save the time taken to do this by using the recovery manager utility program, DFHRMUTL, described in the *CICS Operations and Utilities Guide*.
   - Before a cold start, run DFHRMUTL with **SET_AUTO_START=AUTOCOLD,COLD_COPY** as input parameters. This creates a copy of the global catalog data set that contains only those records needed for a cold start. If the return code from this job step is normal, you can replace the original global catalog with the new copy (taking an archive of the original catalog if you wish). A example of the JCL to do this is contained in the *CICS Operations and Utilities Guide*.
   - Before an initial start, run DFHRMUTL with **SET_AUTO_START=AUTOINIT,COLD_COPY** as input parameters, and follow the same procedure to use the resulting catalog.
9. Allocate your DATA and INDEX data sets on different units, if possible.
10. Consider the use of autoinstalled terminals as a way of improving cold start, even if you do not expect any storage savings. On startup, fewer terminals are installed, thereby reducing the startup time.
11. The RAPOOL system initialization parameter should be set to a value that allows faster autoinstall rates. For a discussion of this, see "Receive-any pool (RAPOOL)" on page 200.
12. Specify the buffer, string, and key length parameters in the LSR pool definition. This reduces the time taken to build the LSR pool, and also reduces the open time for the first file to use the pool.
13. If you have defined performance groups for the CICS system, ensure that all steps preceding the CICS step are also in the same performance group or, at least, have a high enough dispatching priority so as not to delay their execution.

14. The use of DISP=(...,PASS) on any non-VSAM data set used in steps preceding CICS reduces allocation time the next time they are needed. If you do not use PASS on the DD statement, this causes the subsequent allocation of these data sets to go back through the catalog: a time-consuming process.
15. If possible, have one VSAM user catalog with all of the CICS VSAM data sets and use a STEPCAT DD statement to reduce the catalog search time.
16. Keep the number of libraries defined by DFHRPL to a minimum. One large library requires less time to perform the LLACOPY than many smaller libraries.
17. Use of the shared modules in the link pack area (LPA) can help to reduce the time to load the CICS nucleus modules. See the *CICS Transaction Server for OS/390 Installation Guide* for advice on how to install CICS modules in the LPA.
18. CICS does not load programs at startup time for resident programs. The storage area is reserved, but the program is actually loaded on the first access through program control for that program. This speeds startup. The correct way to find a particular program or table in storage is to use the program-control LOAD facility to find the address of the program or table. The use of the LOAD facility physically loads the program into its predefined storage location if it is the first access.

    The use of a PLTPI task to load these programs is one possible technique, but you must bear in mind that the CICS system is not operational until the PLTPI processing is complete, so you should not load every program. Load only what is necessary, or the startup time will appear to increase.

Automatic restart management

Automatic restart management (ARM) is a sysplex-wide integrated restart mechanism that:

- Restarts MVS subsystems in place if they abend (or if notified of a stall condition by a monitor program)
- Restarts all the elements of a workload (for example, CICS TORs, AORs, FORs, DB2, and so on) on another MVS image after an MVS failure
- Restarts a failed MVS image.

You can use the MVS automatic restart manager to implement a sysplex-wide integrated automatic restart mechanism. A sysplex can use ARM and VTAM persistent sessions spread across many TORs in a generic resource set. ARM and VTAM persistent sessions provide good recovery times in the event of a TOR failure, and the TOR restart is reduced because only a fraction of the network has to be rebuilt. You can log on to the generic resource while the failed TOR restarts.

ARM provides faster restart by providing surveillance and automatic restart. The need for operator-initiated restarts, or other automatic restart packages, are eliminated. For more information about MVS automatic restart management, see the *CICS Transaction Server for OS/390 Installation Guide*, and the *OS/390 MVS Setting up a Sysplex* manual, GC28-1779.

MVS automatic restart is available only to non-XRF CICS regions.

Buffer considerations

The number of index levels can be obtained by using the IDCAMS LISTCAT command against a GCD after CICS has been shut down. Because cold start mainly uses sequential processing, it should not require any extra buffers over those automatically allocated when CICS opens the file.

You may wish to increase the number of buffers to improve autoinstall performance. The minimum you should specify is the number suggested above for warm shutdown. This should stop the high-level index being read for each autoinstall.

Note that if you have a large number of terminals autoinstalled, shutdown can fail due to the MXT system initialization parameter being reached or CICS becoming short on storage. To prevent this possible cause of shutdown failure, you should consider putting the CATD transaction in a class of its own to limit the number of concurrent CATD transactions. Also, AIQMAX can be specified to limit the number of devices that can be queued for autoinstall. This protects against abnormal consumption of virtual storage by the autoinstall process, caused as a result of some other abnormal event.

If this limit is reached, the AIQMAX system initialization parameter affects the LOGON, LOGOFF and BIND processing by CICS. CICS requests VTAM to stop passing such requests to CICS. VTAM holds the requests until CICS indicates that it can accept further commands (this occurs when CICS has processed a queued autoinstall request).

Part 5. Appendixes

Appendix A. CICS statistics tables

This appendix provides reference information about CICS statistics. For information about the interpretation of CICS statistics, see "Chapter 5. Using CICS statistics" on page 35.

Interpreting CICS statistics

All five types of CICS statistics record (interval, end-of-day, requested, requested reset, and unsolicited) present information as SMF records. The numbers used to identify each SMF statistics record are given in the DFHSTIDS copybook. Programming information about the formats of CICS statistics records is given in the *CICS Customization Guide*.

Each area of CICS statistics is listed below in the following format:

Statistics area: Statistics type

Brief description, if appropriate.

Name of the assembler DSECT mapping this data.

DFHSTUP name	Field name	Description
DFHSTUP name is the name as it appears on the DFHSTUP report.	**Field name** is the name as it appears in the DSECT referred to above.	**Description** is a brief description of the statistics field. Reset characteristic: Reset characteristic of the statistics field at a statistics interval collection. The values can be: • Not reset • Reset to zero • Reset to 1 • Reset to current values (this applies to peak values only) • An exception to the above (these will be documented).

Statistics areas are listed alphabetically.

Summary report

The Statistics Utility Program (STUP) provides a summary report facility that can be selected using a DFHSTUP control parameter. Information on how to run DFHSTUP is given in the *CICS Operations and Utilities Guide*. When selected, the summary report is placed after all other reports. The DFHSTUP summary report facility summarizes (totals, peaks, and averages) the interval, unsolicited, requested reset and end-of-day statistics on an "applid" by "applid" basis. Requested statistics are not involved in the production of the summary report.

The summary report feature uses all of the appropriate statistic collections contained on the SMF data set. Therefore, depending on when the summary report feature is executed and when the SMF data set was last cleared, summary reports may be produced covering an hour, a week, or any desired period of time. Note that due to the potential magnitude of the summary data, it is not recommended that a summary period extend beyond one year.

Within each of the following sections, the meaning of the summary statistics is given. Because the summary statistics are computed offline by the DFHSTUP utility, the summary statistics are not available to online users. Due to the potential magnitude of the summary data, and due to limited page width, summary data may be represented as a scaled value. For example, if the total number of terminal input messages is 1234567890, this value is shown as 1234M, where 'M' represents millions. Other scaling factors used are 'B' for billions and 'T' for trillions. Scaling is only performed when the value exceeds 99999999, and only then when page width is limited, for example in terminal statistics.

Table 14. Statistics listed in this appendix

Statistic type	DSECT	Page
Autoinstall global statistics	DFHA04DS	341
CICS DB2		
–Global	DFHD2GDS	346
–Resource	DFHD2RDS	353
DBCTL session termination	DFHDBUDS	358
Dispatcher domain		
–Global	DFHDSGDS	360
–TCB	DFHDSGDS	363
Dump domain		
–System global	DFHSDGDS	367
–System resource	DFHSDRDS	367
–Transaction global	DFHTDGDS	370
–Transaction resource	DFHTDRDS	370
Enqueue domain	DFHNQGDS	372
FEPI connection	DFHA23DS	375
FEPI pool	DFHA22DS	376
FEPI target	DFHA24DS	377
File control		
–Resource	DFHA17DS	380
–Data table	DFHA17DS	385
–Performance	DFHA17DS	389
ISC/IRC		
–System entry	DFHA14DS	391
–Mode entry	DFHA20DS	399
–Attach time	DFHA21DS	404
Journalname	DFHLGRDS	405
Log stream	DFHLGSDS	407
LSRpools		
–Resource	DFHA08DS	410
–Data buffer	DFHA08DS	412
–Index buffer	DFHA08DS	414
LSRpool files	DFHA09DS	420
Monitoring		
–Global	DFHMNGDS	421
Program autoinstall	DFHPGGDS	424
Loader domain		

Table 14. Statistics listed in this appendix (continued)

Statistic type	DSECT	Page
–Global	DFHLDGDS	424
Program		
–Resource	DFHLDRDS	436
Recovery manager	DFHRMGDS	439
Statistics domain	DFHSTGDS	445
Storage manager		
–Domain	DFHSMDDS	257
–DSA	DFHSMSDS	450
–Task subpools	DFHSMTDS	456
Table manager	DFHA16DS	458
TCP/IP Services		
–Resource	DFHSORDS	459
Temporary storage	DFHTSGDS	462
Terminal control	DFHA06DS	468
Transaction class	DFHXMCDS	472
Transaction manager		
–Global	DFHXMGDS	476
–Resource	DFHXMRDS	478
Transient data		
–Global	DFHTQGDS	485
–Resource	DFHTQRDS	488
User domain	DFHUSGDS	493
VTAM	DFHA03DS	494

Autoinstall global statistics

This is the DFHSTUP listing for terminals that are connected, while the system is running, by means of the autoinstall facility. These statistics are obtained as **interval**, **end-of-day**, or **requested** statistics. CICS also records **unsolicited** autoinstall statistics, which DFHSTUP prints in a separate report; see "CICS intercommunication facilities" on page 301.

Autoinstall: Local definition statistics

These statistics are available online, and are mapped by the DFHA04DS DSECT.

Table 15. Autoinstall: Local definition statistics

DFHSTUP name	Field name	Description
Autoinstall attempts	A04VADAT	is the number of eligible autoinstall attempts made during the current session of CICS to create terminal entries as users logged on. For an attempt to be considered eligible, CICS and VTAM must not be terminating, autoinstall must be enabled, and the terminal type must be valid for autoinstall (not pipeline, LU6.1, or LU6.2 parallel sessions). Reset characteristic: reset to zero

Table 15. Autoinstall: Local definition statistics (continued)

DFHSTUP name	Field name	Description
Rejected attempts	A04VADRJ	is the number of eligible autoinstall attempts that were subsequently rejected during the current session of CICS. Reasons for rejection can be maximum concurrency value exceeded, invalid bind, the user program has rejected the logon, and so on. If this number is unduly high, check the reasons for rejection.
Deleted attempts	A04VADLO	Reset characteristic: reset to zero is the number of deletions of terminal entries as users logged off during the current session.
Peak concurrent attempts	A04VADPK	Reset characteristic: reset to zero is the highest number of attempts made during the current session to create terminal entries as users logged on at the same time.
Times the peak was reached	A04VADPX	Reset characteristic: reset to current value is the number of times when the highest number of attempts were made during the current session to create terminal entries as users logged on at the same time.
Times SETLOGON HOLD issued	A04VADSH	Reset characteristic: reset to 1 is the number of times that the SETLOGON HOLD command was issued during this run of CICS. CICS issues the VTAM SETLOGON HOLD command when the maximum number of concurrent autoinstall requests allowed (the AIQMAX= system initialization parameter) is exceeded.
Queued logons	A04VADQT	Reset characteristic: reset to zero is the number of attempts that were queued for logon due to delete in progress of the TCTTE for the previous session with the same LU.
Peak of queued logons	A04VADQK	Reset characteristic: reset to zero is the highest number of logons that were queued waiting for TCTTE deletion at any one time. If this is unduly high, consider increasing the delete delay interval parameter of the AILDELAY system initialization parameter.
Times queued peak reached	A04VADQX	Reset characteristic: reset to current value is the number of times this peak was reached. Reset characteristic: reset to 1

Autoinstall: remote definitions - shipped terminal statistics

Table 16. Autoinstall:remote definitions - shipped terminal statistics

DFHSTUP name	Field name	Description
Remote delete interval	A04RDINT	is the currently-specified time delay, in the form **hhmmss**, between invocations of the timeout delete transaction that removes redundant shipped terminal definitions. The value is set either by the DSHIPINT system initialization parameter, or by a subsequent SET DELETSHIPPED command. Reset characteristic: not reset
Remote delete idle time	A04RDIDL	is the currently-specified minimum time, in the form **hhmmss**, that an inactive shipped terminal definition must remain installed in this region, before it becomes eligible for removal by the CICS timeout delete transaction. The value is set either by the DSHIPIDL system initialization parameter, or by a subsequent SET DELETSHIPPED command. Reset characteristic: not reset
Shipped terminals built	A04SKBLT	is the number of shipped remote terminal definitions installed at the start of the recording period, plus the number built during the recording period. This value equates to the sum of “Shipped terminals installed” and “Shipped terminals deleted”. Reset characteristic: reset to number of skeletons installed
Shipped terminals installed	A04SKINS	is the number of shipped remote terminal definitions currently installed in this region. Reset characteristic: not reset
Shipped terminals deleted	A04SKDEL	is the number of shipped remote terminal definitions deleted during the recording period by the TIMEOUT transaction. Reset characteristic: reset to zero
Times interval expired	A04TIEXP	is the number of times the remote delete interval (A04RDINT) expired since the start of the recording period. Reset characteristic: reset to zero
Remote deletes received	A04RDREC	is the number of old-style (pre-CICS/ESA 4.1) remote delete instructions received by this region since the start of the recording period. Reset characteristic: reset to zero
Remote deletes issued	A04RDISS	is the number of old-style (pre-CICS/ESA 4.1) remote delete instructions issued by this region since the start of the recording period. Reset characteristic: reset to zero

Table 16. Autoinstall:remote definitions - shipped terminal statistics (continued)

DFHSTUP name	Field name	Description
Successful remote deletes	A04RDDEL	is the number of shipped terminal definitions deleted from this region because of old-style remote delete instructions, since the start of the recording period.
Total idle count	A04TIDCT	Reset characteristic: reset to zero is the total number of times that all previously used remote terminal definitions (whether deleted from the system or currently in the system) had been idle awaiting reuse. This number does not include the remote terminal definitions currently idle awaiting reuse (see A04CIDCT).
NOT IN THE DFHSTUP REPORT	A04TIDLE	Reset characteristic: reset to zero is the total time (expressed in STCK units) that all previously used remote terminal definitions (whether deleted from the system or currently in the system) had been idle awaiting reuse. This number does not include the remote terminal definitions currently idle awaiting reuse (see A04CIDLE).
Average idle time		Reset characteristic: reset to zero is the average idle time (expressed in STCK units) that all previously used remote terminal definitions (whether deleted from the system or currently in the system) had been idle awaiting reuse. This number does not include the remote terminal definitions currently idle awaiting reuse. This value is calculated offline by DFHSTUP and is, therefore, not accessible through the EXEC CICS COLLECT STATISTICS command.
Maximum idle time	A04TMAXI	Reset characteristic: not reset is the maximum time (expressed in STCK units) for which a previously idle shipped terminal definition had been idle during the recording period. This number does not include the remote terminal definitions currently idle awaiting reuse (A04CMAXI).
NOT IN THE DFHSTUP REPORT	A04CIDCT	Reset characteristic: reset to current value Reset characteristic: Not reset
NOT IN THE DFHSTUP REPORT	A04CIDLE	Reset characteristic: Not reset

Table 16. Autoinstall:remote definitions - shipped terminal statistics (continued)

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	A04CMAXI	Reset characteristic: Not reset

Autoinstall: Summary global statistics

Summary statistics are not available online.

Table 17. Autoinstall: Summary global statistics

DFHSTUP name	Description
Autoinstall attempts	is the total number of eligible autoinstall attempts made during the entire CICS session to create terminal entries as users logged on. For an attempt to be considered eligible, CICS and VTAM must not be terminating, autoinstall must be enabled, and the terminal type must be valid for autoinstall (not pipeline, LU6.1, or LU6.2 parallel sessions).
Rejected attempts	is the total number of eligible autoinstall attempts that were subsequently rejected during the entire CICS session. Reasons for rejection can be maximum concurrency value exceeded, invalid bind, the user program has rejected the logon, and so on. If this number is unduly high, check the reasons for rejection.
Deleted attempts	is the total number of deletions of terminal entries as users logged off during the entire session.
Peak concurrent attempts	is the highest number of attempts made during the entire CICS session to create terminal entries as users logged on at the same time.
Times the peak was reached	is the number of times that the “peak concurrent attempts” value was reached during the entire CICS session.
Times SETLOGON HOLD issued	is the number of times that the SETLOGON HOLD command was issued during the entire run of CICS. CICS issues the VTAM SETLOGON HOLD command when the maximum number of concurrent autoinstall requests allowed (the AIQMAX= system initialization parameter) is exceeded.
Queued logons	is the total number of attempts that were queued for logon due to delete in progress of the TCTTE for the previous session with the same LU.
Peak of queued logons	is the highest number of logons that were queued waiting for TCTTE deletion at any one time. If this is unduly high, consider increasing the delete delay interval parameter of the AILDELAY system initialization parameter.
Times queued peak reached	is the number of times that the “peak of queued logons” value was reached.
Remote delete interval	is the currently-specified time delay, in the form **hhmmss**, between invocations of the timeout delete transaction that removes redundant shipped terminal definitions. The value is set either by the DSHIPINT system initialization parameter, or by a subsequent SET DELETSHIPPED command.
Remote delete idle time	is the currently-specified minimum time, in the form **hhmmss**, that an inactive shipped terminal definition must remain installed in this region, before it becomes eligible for removal by the CICS timeout delete transaction. The value is set either by the DSHIPIDL system initialization parameter, or by a subsequent SET DELETSHIPPED command.
Shipped terminals built	is the number of shipped remote terminal definitions installed at the start of the recording period, plus the number built during the recording period. This value equates to the sum of “Shipped terminals installed” and “Shipped terminals deleted”.

Table 17. Autoinstall: Summary global statistics (continued)

DFHSTUP name	Description
Shipped terminals installed	is the number of shipped remote terminal definitions currently installed in this region.
Shipped terminals deleted	is the number of shipped remote terminal definitions deleted during the recording period by the TIMEOUT transaction.
Times interval expired	is the number of times the remote delete interval (A04RDINT) expired since the start of the recording period.
Remote deletes received	is the number of old-style (pre-CICS/ESA 4.1) remote delete instructions received by this region since the start of the recording period.
Remote deletes issued	is the number of old-style (pre-CICS/ESA 4.1) remote delete instructions issued by this region since the start of the recording period.
Successful remote deletes	is the number of shipped terminal definitions deleted from this region because of old-style remote delete instructions, since the start of the recording period.
Total idle count	is the total number of times that all previously used remote terminal definitions (whether deleted from the system or currently in the system) had been idle awaiting reuse. This number does not include the remote terminal definitions currently idle awaiting reuse (see A04CIDCT).
Total average idle time	is the average idle time (expressed in STCK units) that all previously used remote terminal definitions (whether deleted from the system or currently in the system) had been idle awaiting reuse. This number does not include the remote terminal definitions currently idle awaiting reuse.
Maximum idle time	is the maximum time (expressed in STCK units) for which a previously idle shipped terminal definition had been idle during the recording period. This number does not include the remote terminal definitions currently idle awaiting reuse (A04CMAXI).

CICS DB2

CICS DB2: global statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS DB2CONN command, and are mapped by the DFHD2GDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 18. CICS DB2: global statistics

DFHSTUP name	Field name	Description
DB2 Connection Name	D2G_DB2CONN_NAME	is the name of the installed DB2CONN Reset characteristic: not reset
DB2 Sysid	D2G_DB2_ID	is the name of the DB2 subsystem that the CICS is connected to or will connect to. Reset characteristic: not reset

Table 18. CICS DB2: global statistics (continued)

DFHSTUP name	Field name	Description
DB2 Release	D2G_DB2_RELEASE	is the version and release level of the DB2 subsystem that CICS is connected to. If CICS is not currently connected to DB2 the DSECT field contain nulls.
DB2 Connect Time	D2G_CONNECT_TIME_LOCAL	Reset characteristic: not reset is the local time when CICS connected to DB2. The DFHSTUP report expresses this time as hh:mm:ss; however the DSECT field contains the time as a local store clock (STCK) value.
DB2 Disconnect Time	D2G_DISCONNECT_TIME_LOCAL	Reset characteristic: not reset is the local time when CICS disconnected from DB2. The DFHSTUP report expresses this time as hh:mm:ss; however the DSECT field contains the time as a local store clock (STCK) value. The disconnect time will only be present in DB2CONN unsolicited statistics records produced when the CICS DB2 interface is shutdown, after which the time field is cleared to nulls.
TCB Limit	D2G_TCB_LIMIT	Reset characteristic: not reset is the maximum number of subtask TCBs that can be attached to service DB2 requests.
Current number of TCBs	D2G_TCB_CURRENT	Reset characteristic: not reset is the current number of subtask TCBs attached to service DB2 requests.
Peak number of TCBs	D2G_TCB_HWM	Reset characteristic: not reset is the peak number of subtask TCBs attached to service DB2 requests.
Current number of free TCBs	D2G_TCB_FREE	Reset characteristic: reset to current value (D2G_TCB_CURRENT) is the current number of subtask TCBs without a DB2 thread.
Current number of tasks on the TCB Readyq	D2G_TCB_READYQ_CURRENT	Reset characteristic: not reset is the number of CICS tasks queued waiting for a DB2 subtask TCB to become available.
Peak number of tasks on the TCB Readyq	D2G_TCB_READYQ_HWM	Reset characteristic: not reset is the peak number of CICS tasks queued waiting for a DB2 subtask TCB to become available. Reset characteristic: reset to current value (D2G_TCB_READYQ_CURRENT)

Table 18. CICS DB2: global statistics (continued)

DFHSTUP name	Field name	Description
Pool Thread Plan name	D2G_POOL_PLAN_NAME	is the name of the plan used for the pool. If a dynamic plan exit is being used for the pool this DSECT field will be nulls. Reset characteristic: not reset
Pool Thread Dynamic Planexit name	D2G_POOL_PLANEXIT_NAME	is the name of the dynamic plan exit to be used for the pool. If a static plan is being used for the pool this DSECT field will be nulls. Reset characteristic: not reset
Pool Thread Authtype	D2G_POOL_AUTHTYPE	is the type of id to be used for DB2 security checking for pool threads. If an Authid is being used for pool threads this DSECT field contains nulls. Reset characteristic: not reset
Pool Thread Authid	D2G_POOL_AUTHID	is the static id to be used for DB2 security checking for pool threads. If an Authtype is being used for pool threads this DSECT field contains nulls. Reset characteristic: not reset
Pool Thread Accountrec setting	D2G_POOL_ACCOUNTREC	specifies the frequency of DB2 accounting records to be produced for transactions using pool threads. Reset characteristic: not reset
Pool Thread Threadwait setting	D2G_POOL_THREADWAIT	specifies whether transactions should wait for a pool thread or be abended if the number of active pool threads exceed the pool thread limit. Reset characteristic: not reset
Pool Thread Priority	D2G_POOL_PRIORITY	is the priority of the pool thread subtasks relative to the CICS main task (QR TCB). Reset characteristic: not reset
Number of calls using Pool Threads	D2G_POOL_CALLS	is the number of SQL calls made using pool threads. Reset characteristic: reset to zero
Number of Pool Thread Sign-ons	D2G_POOL_SIGNONS	is the number of DB2 sign-ons performed for pool threads. Reset characteristic: reset to zero
Number of Pool Thread Commits	D2G_POOL_COMMITS	is the number of two phase commits performed for units of work using pool threads. Reset characteristic: reset to zero

Table 18. CICS DB2: global statistics (continued)

DFHSTUP name	Field name	Description
Number of Pool Thread Aborts	D2G_POOL_ABORTS	is the number of units of work using pool threads that were rolled back. Reset characteristic: reset to zero
Number of Pool Thread Single Phases	D2G_POOL_SINGLE_PHASE	is the number of units of work using pool threads that used single phase commit, either because they were read-only UOWs, or because DB2 was the only recoverable resource updated in the UOW. Reset characteristic: reset to zero
Number of Pool Thread Reuses	D2G_POOL_THREAD_REUSE	is the number of times CICS transactions using the pool were able to reuse an already created DB2 thread. This count includes transactions that overflow to the pool to acquire a thread and reuse an existing thread. Reset characteristic: reset to zero
Number of Pool Thread Terminates	D2G_POOL_THREAD_TERM	is the number of terminate thread requests made to DB2 for pool threads. This includes pool threads used by transactions that overflow to the pool. Reset characteristic: reset to zero
Number of Pool Thread Waits	D2G_POOL_THREAD_WAITS	is the number of times all available threads in the pool were busy and a transaction had to wait for a thread to become available. This count includes transactions that overflow to the pool to acquire a thread and have to wait for a pool thread. Reset characteristic: reset to zero
Current Pool Thread Limit	D2G_POOL_THREAD_LIMIT	is the current maximum number of pool threads allowed. Reset characteristic: not reset
Current number of Pool Threads in use	D2G_POOL_THREAD_CURRENT	is the current number of active pool threads. Reset characteristic: not reset
Peak number of Pool Threads in use	D2G_POOL_THREAD_HWM	is the peak number of active pool threads. Reset characteristic: reset to current value (D2G_POOL_THREAD_CURRENT)
Current number of Pool tasks	D2G_POOL_TASK_CURRENT	is the current number of CICS tasks that are using a pool thread. Reset characteristic: not reset

Table 18. CICS DB2: global statistics (continued)

DFHSTUP name	Field name	Description
Peak number of Pool tasks	D2G_POOL_TASK_HWM	is the peak number of CICS tasks that have used a pool thread. Reset characteristic: reset to current value (D2G_POOL_TASK_CURRENT)
Total number of Pool tasks	D2G_POOL_TASK_TOTAL	is the total number of completed tasks that have used a pool thread. Reset characteristic: reset to zero.
Current number of tasks on the Pool Readyq	D2G_POOL_READYQ_CURRENT	is the current number of CICS tasks waiting for a pool thread to become available. Reset characteristic: not reset
Peak number of tasks on the Pool Readyq	D2G_POOL_READYQ_HWM	is the peak number of CICS tasks that waited for a pool thread to become available. Reset characteristic: reset to current value (D2G_POOL_READYQ_CURRENT)
Command Thread Authtype	D2G_COMD_AUTHTYPE	is the type of id to be used for DB2 security checking for command threads. If an Authid is being used for command threads this DSECT field contains nulls. Reset characteristic: not reset
Command Thread Authid	D2G_COMD_AUTHID	is the static id to be used for DB2 security checking for command threads. If an Authtype is being used for command threads this DSECT field contains nulls. Reset characteristic: not reset
Number of calls using Command Threads	D2G_COMD_CALLS	is the number of DB2 commands issued using the DSNC transaction. Reset characteristic: reset to zero
Number of Command Thread Sign-ons	D2G_COMD_SIGNONS	is the number of DB2 sign-ons performed for command threads. Reset characteristic: reset to zero
Number of Command Thread Terminates	D2G_COMD_THREAD_TERM	is the number of terminate thread requests made to DB2 for command threads. Reset characteristic: reset to zero
Number of Command Thread Overflows to Pool	D2G_COMD_THREAD_OVERF	is the number of times a DSNC DB2 command resulted in a pool thread being used because the number of active command threads exceed the command thread limit. Reset characteristic: reset to zero

Table 18. CICS DB2: global statistics (continued)

DFHSTUP name	Field name	Description
Command Thread Limit	D2G_COMD_THREAD_LIMIT	is the current maximum number of command threads allowed. Reset characteristic: not reset
Current number of Command Threads	D2G_COMD_THREAD_CURRENT	is the current number of active command threads. Reset characteristic: not reset
Peak number of Command Threads	D2G_COMD_THREAD_HWM	is the peak number of active command threads. Reset characteristic: reset to current value (D2G_COMD_THREAD_CURRENT)
NOT IN THE DFHSTUP REPORT	D2G_CONNECT_TIME_GMT	is the Greenwich mean time (GMT) when CICS connected to DB2. The DFHSTUP report expresses this time as hh:mm:ss; however the DSECT field contains the time as a GMT store clock (STCK) value. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	D2G_DISCONNECT_TIME_GMT	is the Greenwich mean time (GMT) when CICS disconnected from DB2. The DFHSTUP report expresses this time as hh:mm:ss; however the DSECT field contains the time as a GMT store clock (STCK) value. The disconnect time will only be present in DB2CONN unsolicited statistics records produced when the CICS DB2 interface is shutdown, after which the time field is cleared to nulls. Reset characteristic: not reset

CICS DB2: summary global statistics

Summary statistics are not available online.

Table 19. CICS DB2: summary global statistics

DFHSTUP name	Description
DB2 Connection Name	is the name of the installed DB2CONN
Total DB2 Connection time	is the total amount of time CICS was connected to the DB2 subsystem specified in this DB2CONN. The time is displayed as days:hh:mm:ss.
DB2 Sysid	is the DB2 sysid specified in this DB2CONN. If the sysid has changed, it is the last setting of sysid.
DB2 Release	is the DB2 version and release for this DB2CONN. If the version and release have changed, it is the last setting of version and release.
TCB Limit	is the TCBLIMIT value that was set in the DB2CONN. If the TCBLIMIT has changed, it is the last setting of TCBLIMIT.
Peak number of TCBs	is the peak number of subtask TCBs attached to service DB2 requests

Table 19. CICS DB2: summary global statistics (continued)

DFHSTUP name	Description
Peak number of tasks on the TCB Readyq	is the peak number of CICS tasks queued waiting for a DB2 subtask TCB to become available.
Pool Thread Plan name	is the name of the plan used for the pool. If the plan name has changed, it is the last setting of plan name. If a dynamic plan exit is being used for the pool this DSECT field will be nulls.
Pool Thread Dynamic Planexit name	is the name of the dynamic plan exit to be used for the pool. If the dynamic planexit name has changed, it is the last setting of dynamic planexit name. If static plan is being used for the pool this DSECT field will be nulls.
Pool Thread Authtype	is the type of id to be used for DB2 security checking for pool threads. If the pool thread authtype has changed, it is the last setting of pool thread authtype. If an Authid is being used for pool threads this DSECT field contains nulls.
Pool Thread Authid	is the static id to be used for DB2 security checking for pool threads. If the pool thread authid has changed, it is the last setting of pool thread authid. If an Authtype is being used for pool threads this DSECT field contains nulls.
Pool Thread Accountrec setting	is the frequency of DB2 accounting records to be produced for transactions using pool threads. If the pool thread accountrec setting has changed, it is the last setting of pool thread accountrec.
Pool Thread Threadwait setting	is the setting for whether transactions should wait for a pool thread to be abended if the number of active pool threads reach the pool thread limit. If the pool thread threadwait setting has changed, it is the last setting of pool thread threadwait.
Pool Thread Priority	is the priority of the pool thread subtasks relative to the CICS main task (QR TCB). If the pool thread priority has changed, it is the last setting of pool thread priority.
Total number of calls using Pool Threads	is the total number of SQL calls made using pool threads.
Total number of Pool Thread Sign-ons	is the total number of DB2 sign-ons performed for pool threads.
Total number of Pool Thread Commits	is the total number of two phase commits performed for units of work using pool threads.
Total number of Pool Thread Aborts	is the total number of units of work using pool threads that were rolled back.
Total number of Pool Thread Single Phases	is the total number of units of work using pool threads that used single phase commit, either because they were read-only UOWs, or because DB2 was the only recoverable resource updated in the UOW.
Total number of Pool Thread Reuses	is the total number of times CICS transactions using the pool were able to reuse an already created DB2 thread. This count includes transactions that overflow to the pool to acquire a thread and reuse an existing thread.
Total number of Pool Thread Terminates	is the total number of terminate thread requests made to DB2 for pool threads. This includes pool threads used by transactions that overflow to the pool.
Total number of Pool Thread Waits	is the total number of times all available threads in the pool were busy and a transaction had to wait for a thread to become available. This count includes transactions that overflow to the pool to acquire a thread and have to wait for a pool thread.
Pool Thread Limit	is the thread limit value for the pool. If the pool thread limit has changed, it is the last setting of pool thread limit.
Peak number of Pool Threads in use	is the peak number of active pool threads.

Table 19. CICS DB2: summary global statistics (continued)

DFHSTUP name	Description
Peak number of Pool tasks	is the peak number of CICS tasks that have used a pool thread.
Total number of Pool tasks	is the total number of completed tasks that have used a pool thread.
Peak number of tasks on the Pool Readyq	is the peak number of CICS tasks that waited for a pool thread to become available.
Command Thread Authtype	is the type of id to be used for DB2 security checking for command threads. If the command thread authtype has changed, it is the last setting of command thread authtype. If an Authid is being used for command threads this DSECT field contains nulls.
Command Thread Authid	is the static id to be used for DB2 security checking for command threads. If the command thread authid has changed, it is the last setting of command thread authid. If an Authtype is being used for command threads this DSECT field contains nulls.
Total number of calls using Command Threads	is the total number of DB2 commands issued through the DSNC transaction.
Total number of Command Thread Sign-ons	is the total number of DB2 sign-ons performed for command threads.
Total number of Command Thread Terminates	is the total number of terminate thread requests made to DB2 for command threads.
Total Number of Command Thread Overflows	is the total number of times a DSNC DB2 command resulted in a pool thread being used because the number of active command threads exceed the command thread limit.
Command Thread Limit	is the maximum number of command threads allowed. If the command thread limit has changed, it is the last setting of command thread limit.
Peak number of Command Threads	is the peak number of active command threads.

CICS DB2: resource statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS DB2ENTRY() command and are mapped by the DFHD2RDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

There are three sections in the DFHSTUP report for CICS DB2 (resource) statistics:

- Resource information (see Resource statistics: resource information)
- Request information (see "Resource statistics: request information" on page 354)
- Performance information (see "Resource statistics: performance information" on page 355)

Resource statistics: resource information

The resource information gives details of various attribute settings of each DB2ENTRY.

Table 20. Resource statistics: resource information

DFHSTUP name	Field name	Description
DB2Entry Name	D2R_DB2ENTRY_NAME	is the name of the installed DB2ENTRY
Plan Name	D2R_PLAN_NAME	Reset characteristic: not reset is the name of the plan used for this DB2ENTRY. If a dynamic plan exit is being used for the DB2Entry this DSECT field will be nulls.
PlanExit name	D2R_PLANEXIT_NAME	Reset characteristic: not reset is the name of the dynamic planexit to be used for this DB2ENTRY. If a static plan is being used for the DB2ENTRY this DSECT field will be nulls.
Auth id	D2R_AUTHID	Reset characteristic: not reset is the static id to be used for DB2 security checking for this DB2ENTRY. If an Authtype is being used for the DB2ENTRY this DSECT field contains nulls.
Auth type	D2R_AUTHTYPE	Reset characteristic: not reset is the type of id to be used for DB2 security checking for this DB2ENTRY. If an Authid is being used for the DB2ENTRY this DSECT field contains nulls.
Account records	D2R_ACCOUNTREC	Reset characteristic: not reset specifies the frequency of DB2 accounting records to be produced for transactions using this DB2ENTRY.
Thread Wait	D2R_THREADWAIT	Reset characteristic: not reset specifies whether transactions should wait for a thread, abend or overflow to the pool, if the number of active threads for this DB2ENTRY exceeds its thread limit.
Thread Prty	D2R_PRIORITY	Reset characteristic: not reset is the priority of the DB2ENTRY thread subtasks relative to the CICS main task (QR TCB).
		Reset characteristic: not reset

Resource statistics: request information

The request information gives details of how many requests of various types have been performed against each DB2ENTRY.

Table 21. Resource statistics: request information

DFHSTUP name	Field name	Description
DB2Entry Name	D2R_DB2ENTRY_NAME	is the name of the installed DB2ENTRY
Call Count	D2R_CALLS	Reset characteristic: not reset is the number of SQL calls made using this DB2ENTRY.
Sign-on Count	D2R_SIGNONS	Reset characteristic: reset to zero is the number of DB2 sign-ons performed for this DB2ENTRY.
Commit Count	D2R_COMMITS	Reset characteristic: reset to zero is the number of two phase commits performed for units of work using this DB2ENTRY.
Abort Count	D2R_ABORTS	Reset characteristic: reset to zero is the number of units of work using this DB2ENTRY that were rolled back.
Single Phase	D2R_SINGLE_PHASE	Reset characteristic: reset to zero is the number of units of work using the DB2ENTRY that used single-phase commit, either because they were read-only UOWs, or because DB2 was the only recoverable resource updated in the UOW.
Thread Reuse	D2R_THREAD_REUSE	Reset characteristic: reset to zero is the number of times CICS transactions using the DB2ENTRY were able to reuse an already created DB2 thread.
Thread Terms	D2R_THREAD_TERM	Reset characteristic: reset to zero is the number of terminate thread requests made to DB2 for threads of this DB2ENTRY.
Thread Waits/Overflows	D2R_THREAD_WAIT_OR_OVERF	Reset characteristic: reset to zero is the number of times all available threads in the DB2ENTRY were busy and a transaction had to wait for a thread to become available, or overflow to the pool and use a pool thread instead.
		Reset characteristic: reset to zero

Resource statistics: performance information

The performance information gives details of Thread information for each DB2ENTRY.

Table 22. Resource statistics: performance information

DFHSTUP name	Field name	Description
DB2ENTRY Name	D2R_DB2ENTRY_NAME	is the name of the installed DB2ENTRY
Thread Limit	D2R_THREAD_LIMIT	Reset characteristic: not reset is the current maximum number of threads allowed for the DB2ENTRY.
Thread Current	D2R_THREAD_CURRENT	Reset characteristic: not reset is the current number of active threads for this DB2ENTRY.
Thread HWM	D2R_THREAD_HWM	Reset characteristic: not reset is the peak number of active threads for this DB2ENTRY.
Pthread Limit	D2R_PTHREAD_LIMIT	Reset characteristic: reset to current value (D2R_THREAD_CURRENT) is the current maximum number of protected threads allowed for this DB2ENTRY.
PThread Current	D2R_PTHREAD_CURRENT	Reset characteristic: not reset is the current number of protected threads for this DB2ENTRY.
Pthread HWM	D2R_PTHREAD_HWM	Reset characteristic: not reset is the peak number of protected threads for this DB2ENTRY.
Task Current	D2R_TASK_CURRENT	Reset characteristic: reset to current value (D2R_PTHREAD_CURRENT) is the current number of CICS tasks that are using this DB2ENTRY.
Task HWM	D2R_TASK_HWM	Reset characteristic: not reset is the peak number of CICS tasks that have used this DB2ENTRY.
Task Total	D2R_TASK_TOTAL	Reset characteristic: reset to current value (D2R_TASK_CURRENT) is the total number of completed tasks that have used this DB2ENTRY.
Readyq Current	D2R_READYQ_CURRENT	Reset characteristic: reset to zero. is the current number of CICS tasks waiting for a thread to become available on this DB2ENTRY.
Readyq HWM	D2R_READYQ_HWM	Reset characteristic: not reset is the peak number of CICS tasks that waited for a thread to become available on this DB2ENTRY.
		Reset characteristic: reset to current value (D2R_READYQ_CURRENT)

CICS DB2: summary resource statistics

There are three sections in the DFHSTUP summary report for CICS-DB2 (resource) statistics:

- Resource information (see Summary resource statistics: resource information)
- Request information (see "Summary resource statistics: request information")
- Performance information (see "Summary resource statistics: performance information" on page 358)

Summary statistics are not available online.

Summary resource statistics: resource information

The resource information gives details of various attribute settings of each DB2ENTRY.

Table 23. Summary resource statistics: resource information

DFHSTUP name	Description
DB2Entry Name	is the name of the installed DB2ENTRY.
Plan Name	is the name of the plan used for this DB2ENTRY. If a dynamic plan exit is being used for the DB2Entry this DSECT field will be nulls. If the plan name changed, it is the last setting of plan name.
PlanExit name	is the name of the dynamic planexit to be used for this DB2ENTRY. If the planexit name has changed, it is the last setting of PlanExit name. If a static plan is being used for the DB2ENTRY this DSECT field will be nulls.
Auth id	is the static id to be used for DB2 security checking for this DB2ENTRY. If the Auth id changed, it is the last setting of Auth id. If an Authtype is being used for the DB2ENTRY this DSECT field contains nulls.
Auth type	is the type of id to be used for DB2 security checking for this DB2ENTRY. If the Auth type changed, it is the last setting of Auth type. If an Authid is being used for the DB2ENTRY this DSECT field contains nulls.
Account records	specifies the frequency of DB2 accounting records to be produced for transactions using this DB2ENTRY. If the frequency changed, it is the last frequency setting.
Thread Wait	specifies whether transactions should wait for a thread, abend, or overflow to the pool, if the number of active threads for this DB2ENTRY exceeds its thread limit. If the threadwait changed, it is the last setting of threadwait.
Thread Prty	is the priority of the DB2ENTRY thread subtasks relative to the CICS main task (QR TCB). If the priority changed, it is the last setting of priority.

Summary resource statistics: request information

The request information gives details of how many requests of various types have been performed against each DB2ENTRY.

Table 24. Summary resource statistics: request information

DFHSTUP name	Description
DB2Entry Name	is the name of the installed DB2ENTRY
Call Count	is the total number of SQL calls made using this DB2ENTRY.
Sign-on Count	is the total number of DB2 sign-ons performed for this DB2ENTRY.
Commit Count	is the total number of two phase commits performed for units of work using this DB2ENTRY.
Abort Count	is the total number of units of work using this DB2ENTRY that were total back.

Table 24. Summary resource statistics: request information (continued)

DFHSTUP name	Description
Single Phase	is the total number of units of work using the DB2ENTRY that used single phase commit, either because they were read-only UOWs, or because DB2 was the only recoverable resource updated in the UOW.
Thread Reuse	is the total number of times CICS transactions using the DB2ENTRY were able to reuse an already created DB2 thread.
Thread Terms	is the total number of terminate thread requests made to DB2 for threads of this DB2ENTRY.
Thread Waits/Overflows	is the total number of times all available threads in the DB2ENTRY were busy and a transaction had to wait for a thread to become available, or overflow to the pool and use a pool thread instead.

Summary resource statistics: performance information

The performance information gives details of thread information for each DB2ENTRY.

Table 25. Summary resource statistics: performance information

DFHSTUP name	Description
DB2ENTRY Name	is the name of the installed DB2ENTRY
Thread Limit	is the maximum number of threads allowed for the DB2ENTRY. If the value changed, it is the last setting of Thread limit.
Thread HWM	is the peak number of active threads for this DB2ENTRY.
Pthread Limit	is the maximum number of protected threads allowed for this DB2ENTRY. If the value changed, it is the last setting of Pthread limit.
Pthread HWM	is the peak number of protected threads for this DB2ENTRY.
Task HWM	is the peak number of CICS tasks that have used this DB2ENTRY.
Task Total	is the total number of completed tasks that have used this DB2ENTRY.
Readyq HWM	is the peak number of CICS tasks that waited for a thread to become available on this DB2ENTRY.

DBCTL session termination

DBCTL statistics are of the **unsolicited** type only. They appear on a separate report to the other types of CICS statistics.

The DBCTL statistics exit DFHDBSTX is invoked by the CICS adapter (DFHDBAT), and CICS statistics information is collected by the statistics domain whenever DBCTL is disconnected as a result of:

- An orderly or immediate disconnection of the DBCTL using the menu transaction CDBC
- An orderly termination of CICS.

Note: If there is an immediate shutdown or abend of CICS, the latest CICS-DBCTL session statistics are lost. The function of DFHDBSTX is to invoke the statistics domain supplying the data that has been returned from the database resource adapter (DRA) relating to the individual CICS-DBCTL session.

CICS termination statistics that contain the number of DL/I calls by type, issued against each DL/I database, are not produced by CICS in the DBCTL environment. DBCTL produces this type of information.

For more information about CICS-DBCTL statistics, see the *CICS IMS Database Control Guide*.

DBCTL: global statistics

These statistics are mapped by the DFHDBUDS DSECT.

Table 26. DBCTL: Global statistics

DFHSTUP name	Field name	Description
CICS DBCTL session number	STADSENO	is the number of the CICS-DBCTL session and is incremented every time you connect and disconnect. Reset characteristic: not reset
DBCTL identifier	STATDBID	is the name of the DBCTL session. Reset characteristic: not reset
DBCTL RSE name	STARSEN	is the name of the DBCTL recoverable service element (RSE). Reset characteristic: not reset
Time CICS connected to DBCTL	STALCTIM	is the time when CICS was connected to DBCTL. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals* at local time; however, the DSECT field contains the time as a local store clock (STCK) value. Reset characteristic: not reset
Time CICS disconnected from DBCTL	STALDTIM	is the time when CICS was disconnected from DBCTL. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals* at local time; however, the DSECT field contains the time as a local store clock (STCK) value. Reset characteristic: not reset
NOT IN DFHSTUP REPORT	STACTIME	is the time when CICS was connected to DBCTL. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals* at GMT; however, the DSECT field contains the time as a GMT store clock (STCK) value. Reset characteristic: not reset
NOT IN DFHSTUP REPORT	STADTIME	is the time when CICS was disconnected from DBCTL. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals* at GMT; however, the DSECT field contains the time as a GMT store clock (STCK) value. Reset characteristic: not reset
Minimum number of threads	STAMITHD	is the minimum value specified in the DRA startup parameter table. Reset characteristic: not reset
Maximum number of threads	STAMATHD	is the maximum value specified in the DRA startup parameter table. Reset characteristic: not reset

Table 26. DBCTL: Global statistics (continued)

DFHSTUP name	Field name	Description
Times minimum threads hit	STANOMITHD	is the number of times the CICS-DBCTL session has hit the minimum thread value. Reset characteristic: not reset
Times maximum threads hit	STANOMATHD	is the number of times the CICS-DBCTL session has hit the maximum thread value. Reset characteristic: not reset
Elapsed time at maximum threads	STAELMAX	is the elapsed time, expressed as *hours:minutes:seconds.decimals*, for which the CICS-DBCTL session is running at the maximum thread value. Reset characteristic: none
Peak number of threads	STAHIWAT	is the highest number of threads used throughout the CICS-DBCTL session. Reset characteristic: not reset
Successful PSB schedules	STAPSBSU	is the number of times the CICS-DBCTL session has successfully scheduled a program specification block (PSB). Reset characteristic: not reset

DBCTL: summary global statistics

Summary statistics are not available online.

Table 27. DBCTL: Summary global statistics

DFHSTUP name	Description
CICS DBCTL session number	is the number of the CICS-DBCTL session.
DBCTL identifier	is the name of the DBCTL session.
DBCTL RSE name	is the name of the DBCTL recoverable service element (RSE).
Minimum number of threads	is the minimum value specified in the DRA startup parameter table.
Maximum number of threads	is the maximum value specified in the DRA startup parameter table.
Times minimum threads hit	is the total number of times the CICS-DBCTL session has hit the minimum thread value.
Times maximum threads hit	is the total number of times the CICS-DBCTL session has hit the maximum thread value.
Elapsed time at maximum threads	is the elapsed time, expressed as *days-hours:minutes:seconds.decimals*, for which the CICS-DBCTL session is running at the maximum thread value.
Peak number of threads	is the highest number of threads used throughout the CICS-DBCTL session.
Successful PSB schedules	is the total number of times the CICS-DBCTL session has successfully scheduled a program specification block (PSB).

Dispatcher domain

Dispatcher domain: Global statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS DISPATCHER command, and are mapped by the DFHDSGDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS

command, see the *CICS System Programming Reference* manual.

Table 28. Dispatcher domain: Global statistics

DFHSTUP name	Field name	Description
Dispatcher domain start date and time	DSGLSTRT	is the date and time at which the dispatcher started. This value can be used as an approximate time at which CICS started. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value in local time. Reset characteristic: not reset
NOT IN DFHSTUP REPORT	DSGSTART	is the time at which the dispatcher started. This value can be used as an approximate time at which CICS started. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value in GMT. Reset characteristic: not reset
Address Space CPU Time	DSGEJST	is the total CPU time for all TCBs in this address space, accumulated during the interval. Reset characteristic: reset to zero
Address Space SRB Time	DSGSRBT	is the total CPU time for all service request blocks (SRB) executed in this address space, accumulated during the interval. Reset characteristic: reset to zero
Current number of tasks	DSGCNT	is the current number of tasks in the system. This figure includes all system tasks and all user tasks. Reset characteristic: not reset
Peak number of tasks	DSGPNT	is the peak value of the number of tasks concurrently in the system. Reset characteristic: reset to current value
Current ICV time (msec)	DSGICVT	is the ICV time value (expressed in *milliseconds*) specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM TIME(value) or EXEC CICS SET SYSTEM TIME(fullword binary data-value) commands. Reset characteristic: not reset
Current ICVR time (msec)	DSGICVRT	is the ICVR time value (expressed im milliseconds) specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM TIME(value) or EXEC CICS SET SYSTEM TIME(fullword binary data-value) commands. Reset characteristic: not reset

Table 28. Dispatcher domain: Global statistics (continued)

DFHSTUP name	Field name	Description
Current ICVTSD time (msec)	DSGICVSD	is the ICVTSD time value (expressed in *milliseconds*) specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM SCANDELAY(value) or EXEC CICS SET SYSTEM SCANDELAY(fullword binary data-value) commands.
Current PRTYAGE time (msec)	DSGPRIAG	Reset characteristic: not reset is the PRTYAGE time value (expressed in milliseconds) specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM AGING(value) or EXEC CICS SET SYSTEM AGING(fullword binary data-value) commands.
Current Max Open TCB limit (MAXOPENTCBS)	DSGMAXOP	Reset characteristic: not reset is the MAXOPENTCBs value specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM MAXOPENTCBS(value) or EXEC CICS SET SYSTEM MAXOPENTCBS (fullword binary data-value) commands
Current Open TCBs	DSGCNUAT	Reset characteristic: not reset is the current number of CICS TCBs attached in an open mode.
Peak Open TCBs	DSGPNUAT	Reset characteristic: not reset is the peak number of CICS TCBs attached in an open mode.
Current Open TCBs in use	DSGCNUUS	Reset characteristic: reset to current value is the current number of CICS TCBs attached in an open mode and being used.
Peak Open TCBs in use	DSGPNUUS	Reset characteristic: not reset is the peak number of CICS TCBs used that were attached in an open mode.
Times at Max Open TCB Limit	DSGNTCBL	Reset characteristic: reset to current value is the number of times the system reached the MAXOPENTCBS limit.
Total TCB Attaches delayed by MAXOPENTCBS	DSGTOTNW	Reset characteristic: reset to zero is the total number of TCB attaches delayed because the system reached the MAXOPENTCBS limit.
Total MAXOPENTCBS delay time	DSGTOTWL	Reset characteristic: reset to zero is the total time that open mode TCBs were delayed because the system had reached the MAXOPENTCBS limit.
		Reset characteristic: reset to zero

Table 28. Dispatcher domain: Global statistics (continued)

DFHSTUP name	Field name	Description
Current Attaches delayed by MAXOPENTCBS	DSGCURNW	is the current number of TCB attaches that are currently delayed because the system is at MAXOPENTCBS limit. Reset characteristic: not reset
Current MAXOPENTCBS delay time	DSGCURWT	is the current delay time for the TCB attaches that are currently delayed because the system is at MAXOPENTCBS limit. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	DSGASIZE	is the current number of CICS TCB modes in which the CICS dispatcher is managing MVS task control blocks (TCBs) in the system under which the CICS dispatcher runs. Reset characteristic: not reset

Dispatcher domain: TCB statistics

The following fields are mapped by the DSGTCB DSECT within the DFHDSGDS DSECT. The DSGTCB DSECT is repeated for each mode of TCB in CICS (DSGASIZE).

For a list of modes of TCB, see "Dispatcher statistics" on page 43.

Table 29. Dispatcher domain: TCB statistics

DFHSTUP name	Field name	Description
Mode	DSGTGBM	is the name of the CICS Dispatcher TCB mode, either QR, RO, CO, SZ, RP, FO, SL, SO, J8, L8 or S8.
Current TCBs	DSGTCBCA	Reset characteristic: not reset is the current number of MVS TCBs attached in this CICS dispatcher TCB mode.
MVS Waits	DSGSYSW	Reset characteristic: not reset is the number of MVS waits which occurred on this TCB.
Accum Time in MVS wait	DSGTWT	Reset characteristic: reset to zero is the accumulated real time that the CICS region was in an MVS wait, that is, the total time used between an MVS wait issued by the dispatcher and the return from the MVS wait. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value. Reset characteristic: reset to zero

Table 29. Dispatcher domain: TCB statistics (continued)

DFHSTUP name	Field name	Description
Accum Time Dispatched	DSGTDT	is the accumulated real time that this TCB has been dispatched by MVS, that is, the total time used between an MVS wait issued by the dispatcher and the subsequent wait issued by the dispatcher. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value. Reset characteristic: reset to zero
NOT IN THE DFHSTUP REPORT	DSGTCT	is the accumulated CPU time taken for this DS task, that is, the processor time used by this TCB while executing the default dispatcher task (DSTCB). The DSECT field contains the time as a store clock (STCK) value. Reset characteristic: reset to zero
Accum CPU Time / TCB	DSGACT	is the accumulated CPU time taken for this TCB, that is, the total time that this TCB has been in execution. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value. Reset characteristic: reset to zero

Dispatcher domain: TCB statistics - Modes

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS DISPATCHER command, and are mapped by the DFHDSGDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 30. Dispatcher domain: TCB statistics - Modes

DFHSTUP name	Field name	Description
Mode	DSGTCBNM	is the name of the CICS Dispatcher TCB mode, either QR, RO, CO, SZ, RP, FO, SL, SO, J8, L8 or S8. Reset characteristic: not reset
Current TCBs	DSGTCBCA	is the current number of MVS TCBs attached in this CICS dispatcher TCB mode. Reset characteristic: not reset
Peak TCBs	DSGTCBPA	is the peak number of MVS TCBs attached in this CICS dispatcher TCB mode. Reset characteristic: reset to current value
TCBs attached	DSGNTCBA	is the number of MVS TCBs that have been attached in this CICS dispatcher TCB mode. Reset characteristic: reset to zero

Table 30. Dispatcher domain: TCB statistics - Modes (continued)

DFHSTUP name	Field name	Description
Detached Unclean	DSGTCBDU	is the number of MVS TCBs that have been or are in the process of being detached from this CICS dispatcher TCB mode because the CICS transaction that was associated with the TCB has abended. Reset characteristic: reset to zero
Detached Stolen	DSGTCBDS	is the number of MVS TCBs that have been or are in the process of being stolen from this CICS dispatcher TCB mode because it is required by another TCB mode. Reset characteristic: reset to zero
Detached Other	DSGTCBDO	is the number of MVS TCBs that have been or are in the process of being detached from this CICS dispatcher TCB mode. For example, MAXOPENTCBS has been lowered of there are too many TCBs attached in relation to the number of TCBs in use. Reset characteristic: reset to zero
TCB Steals	DSGTCBST	is the number of MVS TCBs that have been stolen from other TCB modes. Reset characteristic: reset to zero

Dispatcher domain: Summary global statistics

Summary statistics are not available online.

Table 31. Dispatcher domain: Summary global statistics

DFHSTUP name	Description
Dispatcher start date and time	is the date and time at which the CICS dispatcher started. This value can be used as an approximate date and time at which CICS started. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals* at the local time); however, the DSECT field contains the time as a local store clock (STCK) value.
Address space CPU time	is the total CPU time taken by the CICS address space. The DFHSTUP report expresses this as *hours:minutes:seconds.decimals*
Address space SRB time	is the total SRB time taken by the CICS address space. The DFHSTUP report expresses this as *hours:minutes:seconds.decimals*
Peak number of tasks	is the peak number of tasks concurrently in the system.
Peak ICV time (msec)	is the peak ICV time value (expressed in *milliseconds*) specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM TIME(value) or EXEC CICS SET SYSTEM TIME(fullword binary data-value) commands.
Peak ICV time (msec)	is the peak ICV time value (expressed in *milliseconds*) specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM RUNWAY(value) or EXEC CICS SET SYSTEM RUNWAY (fullword binary data-value) commands.
Peak ICVTSD time (msec)	is the peak ICVTSD time value (expressed in *milliseconds*) specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM SCANDELAY(value) or EXEC CICS SET SYSTEM SCANDELAY(fullword binary data-value) commands.

Table 31. Dispatcher domain: Summary global statistics (continued)

DFHSTUP name	Description
Peak PRTYAGE time (msec)	is the peak PRTYAGE time value (expressed in *milliseconds*) specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM AGING(value) or EXEC CICS SET SYSTEM AGING(fullword binary data-value) commands.
Max open TCB limit (MAXOPENTCBS)	is the last MAXOPENTCBS value (expressed as the number of open TCBs) that was specified in the SIT, or as an override, or changed dynamically using the CEMT SET SYSTEM MAXOPENTCBS(value) or EXEC CICS SET SYSTEM MAXOPENTCBS(fullword binary data-value) commands.
Peak open TCBs in use	is the peak number of open TCBs in use reached in the system.
Times at max open TCB limit	is the total number of times the MAXOPENTCBS limit has been reaches.
Total TCB attaches delayed by MAXOPENTCBS	is the total number of TCB attaches that have been delayed due to the MAXOPENTCBS limit being reached.
Total MAXOPENTCBS delay time	is the total time spent waiting by those tasks that were delayed due to the MAXOPENTCBS limit being reached.
Average MAXOPENTCBS delay time	is the average time spent waiting by those tasks that were delayed due to the MAXOPENTCBS limit being reached.

Dispatcher domain: Summary TCB statistics

The following statistics are repeated for each of the modes of TCB.

For a list of modes of TCB, see "Dispatcher statistics" on page 43.

Table 32. Dispatcher domain: Summary TCB statistics

DFHSTUP name	Description
Mode	is the name of the CICS Dispatcher TCB mode, either QR, RO, CO, SZ, RP, FO, SL, SO, J8, L8 or S8.
Peak TCBs	is the peak number of MVS TCBs attached in this CICS dispatcher TCB mode.
MVS Waits	is the total number of MVS waits which occurred on this TCB mode.
Total Time in MVS wait	is the total real time that the TCBs in this mode were in an MVS wait. The DFHSTUP report expresses this time as *days-hours:minutes:seconds.decimals*.
Total Time Dispatched	is the total real time that the TCBs in this mode were dispatched by MVS. The DFHSTUP report expresses this time as *days-hours:minutes:seconds.decimals*.
Total CPU Time / TCB	is the total CPU time taken for the TCBs in this mode. The DFHSTUP report expresses this time as *days-hours:minutes:seconds.decimals*.

Dispatcher domain: Summary TCB statistics - Modes

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS DISPATCHER command, and are mapped by the DFHDSGDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 33. Dispatcher domain: TCB statistics - Modes

DFHSTUP name	Field name	Description
Mode	DSGTCBNM	is the name of the CICS Dispatcher TCB mode, either QR, RO, CO, SZ, RP, FO, SL, SO, J8, L8 or S8. Reset characteristic: not reset
Current TCBs	DSGTCBCA	is the current number of MVS TCBs attached in this CICS dispatcher TCB mode. Reset characteristic: not reset
Peak TCBs	DSGTCBPA	is the peak number of MVS TCBs attached in this CICS dispatcher TCB mode. Reset characteristic: reset to current value
TCBs attached	DSGNTCBA	is the number of MVS TCBs that have been attached in this CICS dispatcher TCB mode. Reset characteristic: reset to zero
Detached Unclean	DSGTCBDU	is the number of MVS TCBs that have been or are in the process of being detached from this CICS dispatcher TCB mode because the CICS transaction that was associated with the TCB has abended. Reset characteristic: reset to zero
Detached Stolen	DSGTCBDS	is the number of MVS TCBs that have been stolen or are in the process of being stolen from this CICS dispatcher TCB mode because it is required by another TCB mode. Reset characteristic: reset to zero
Detached Other	DSGTCBDO	is the number of MVS TCBs that have been detached or are in the process of being detached from this CICS dispatcher TCB mode. For example, MAXOPENTCBS has been lowered of there are too many TCBs attached in relation to the number of TCBs in use. Reset characteristic: reset to zero
TCB Steals	DSGTCBST	is the number of MVS TCBs that have been stolen from other TCB modes. Reset characteristic: reset to zero

Dump domain

The dump domain collects global and resource statistics for both system and transaction dumps which occur during the CICS run.

System dumps

Dump domain: system dump global statistics

These statistics fields contain the global data collected by the dump domain for system dumps.

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS SYSDUMPCODE command, and are mapped by the DFHSDGDS

DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 34. Dump domain: System dump global statistics

DFHSTUP name	Field name	Description
Dumps taken	SYS_DUMPS_TAKEN	is the number of system dumps taken by the whole system during the present run of CICS. This number does not include suppressed dumps. A set of related dumps may be taken across the sysplex if the dump code includes the RELATED option. In this case, the count is incremented by one for the CICS system which initiated the dump. The number is unchanged for all other CICS systems even if they have issued a dump as part of the related request. Reset characteristic: reset to zero
Dumps suppressed	SYS_DUMPS_SUPPR	is the number of system dumps, requested from the dump domain by CICS or by a user, which were suppressed by one of: • A user exit • The dump table • A global system dump suppression. Reset characteristic: reset to zero

Dump domain: System dump resource statistics

These statistics fields contain the data collected by the dump domain for system dumps, by dump code. They are available online, and are mapped by the DFHSDRDS DSECT.

Table 35. Dump domain: system dump resource statistics

DFHSTUP name	Field name	Description
Dumpcode	SDRCODE	is the system dump code. This code is a CICS message number with the DFH prefix and the action code suffix (if any) removed. For guidance information about CICS messages, see the *CICS Messages and Codes* manual. Reset characteristic: not reset
Dumps	SDRSTKN	is the number of system dumps taken for the dump code identified in the Dumpcode (SDRCODE) field. A set of related dumps may be taken across the sysplex if the dump code includes the RELATED option. In this case, the count is incremented by one for the CICS system which initiated the dump. The number is unchanged for all other CICS systems even if they have issued a dump as part of the related request. Reset characteristic: reset to zero

Table 35. Dump domain: system dump resource statistics (continued)

DFHSTUP name	Field name	Description
Dumps suppressed	SDRSSUPR	is the number of system dumps, for the dump code identified in the Dumpcode (SDRCODE) field, which were suppressed by one of: • A user exit • The dump table • A global system dump suppression. Reset characteristic: reset to zero
NOT IN THE DFHSTUP REPORT	SDRTTKN & SDRTSUPR	These fields are always zero. They exist here only for compatibility with the transaction dump statistics record format. A transaction dump can force a system dump to be taken as well (it is an option in the transaction dump table), but a system dump cannot force a transaction dump to be taken. Reset characteristic: not applicable

Dump domain: summary system dump global statistics

Summary statistics are not available online.

Table 36. Dump domain: Summary system dump global statistics

DFHSTUP name	Description
Dumps taken	is the total number of system dumps taken by the whole system during the entire run of CICS. This number does not include suppressed dumps. A set of related dumps may be taken across the sysplex if the dump code includes the RELATED option. In this case, the count is incremented by one for the CICS system which initiated the dump. The number is unchanged for all other CICS systems even if they have issued a dump as part of the related request.
Dumps suppressed	is the total number of system dumps, requested from the dump domain by CICS or by a user, which were suppressed by one of: • A user exit • The dump table • A global system dump suppression.

Dump domain: summary system dump resource statistics

Summary statistics are not available online.

Table 37. Dump domain: Summary system dump resource statistics

DFHSTUP name	Description
Dumpcode	is the system dump code. This code is a CICS message number with the DFH prefix and the action code suffix (if any) removed. For guidance information about CICS messages, see the *CICS Messages and Codes* manual.

Table 37. Dump domain: Summary system dump resource statistics (continued)

DFHSTUP name	Description
Dumps	is the total number of system dumps taken for the dump code identified in the Dumpcode field. A set of related dumps may be taken across the sysplex if the dump code includes the RELATED option. In this case, the count is incremented by one for the CICS system which initiated the dump. The number is unchanged for all other CICS systems even if they have issued a dump as part of the related request.
Dumps suppressed	is the total number of system dumps, for the dump code identified in the Dumpcode field, which were suppressed by one of: • A user exit • The dump table • A global system dump suppression.

Transaction dumps

Dump domain: transaction dump global statistics

These statistics fields contain the global data collected by the dump domain for transaction dumps.

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS TRANDUMPCODE command and are mapped by the DFHTDGDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 38. Dump domain: transaction dump global statistics

DFHSTUP name	Field name	Description
Dumps taken	TRANS_DUMP_TAKEN	is the number of transaction dumps taken by the whole system during the present run of CICS. This number does not include suppressed dumps. Reset characteristic: reset to zero
Dumps suppressed	TRANS_DUMP_SUPP	is the number of transaction dumps, requested from the dump domain by CICS or by a user, which were suppressed by one of: • A user exit • The dump table. Reset characteristic: reset to zero

Dump domain: transaction dump resource statistics

These statistics fields contain the data collected by the dump domain for transaction dumps, by dump code. They are available online, and are mapped by the DFHTDRDS DSECT.

Table 39. Dump domain: transaction dump resource statistics

DFHSTUP name	Field name	Description
Dumpcode	TDRCODE	is the transaction dump code. For guidance information about transaction abend codes, see the *CICS Messages and Codes* manual. Reset characteristic: not reset

Table 39. Dump domain: transaction dump resource statistics (continued)

DFHSTUP name	Field name	Description
Dumps	TDRTTKN	is the number of transaction dumps taken for the dump code identified in the Dumpcode (TDRCODE) field. Reset characteristic: reset to zero
Dumps suppressed	TDRTSUPR	is the number of transaction dumps suppressed for the dump code identified in the Dumpcode (TDRCODE) field. Reset characteristic: reset to zero
System dumps	TDRSTKN	is the number of system dumps forced by the transaction dump identified in the Dumpcode (TDRCODE) field. A set of related dumps may be taken across the sysplex if the dump code includes the RELATED option. In this case, the count is incremented by one for the CICS system which initiated the dump. The number is unchanged for all other CICS systems even if they have issued a dump as part of the related request. Reset characteristic: reset to zero
System dumps suppressed	TDRSSUPR	is the number of system dumps, forced by the transaction dump identified in the Dumpcode (TDRCODE) field, which were suppressed by one of: • A user exit • The transaction dump table • A global system dump suppression. Reset characteristic: reset to zero

Dump domain: summary transaction dump global statistics

Summary statistics are not available online.

Table 40. Dump domain: Summary transaction dump global statistics

DFHSTUP name	Description
Dumps taken	is the total number of transaction dumps taken by the whole system during the entire run of CICS. This number does not include suppressed dumps.
Dumps suppressed	is the total number of transaction dumps, requested from the dump domain by CICS or by a user, which were suppressed by one of: • A user exit • The dump table.

Dump domain: summary transaction dump resource statistics

Summary statistics are not available online.

Table 41. Dump domain: Summary transaction dump resource statistics

DFHSTUP name	Description
Dumpcode	is the transaction dump code. For guidance information about transaction abend codes, see the *CICS Messages and Codes* manual.

Table 41. Dump domain: Summary transaction dump resource statistics (continued)

DFHSTUP name	Description
Dumps	is the total number of transaction dumps taken for the dump code identified in the Dumpcode field.
Dumps suppressed	is the total number of transaction dumps suppressed for the dump code identified in the Dumpcode field.
System dumps	is the total number of system dumps forced by the transaction dump identified in the Dumpcode field. A set of related dumps may be taken across the sysplex if the dump code includes the RELATED option. In this case, the count is incremented by one for the CICS system which initiated the dump. The number is unchanged for all other CICS systems even if they have issued a dump as part of the related request.
System dumps suppressed	is the total number of system dumps, forced by the transaction dump identified in the Dumpcode field, which were suppressed by one of: • A user exit • The transaction dump table • A global system dump suppression.

Enqueue domain

The enqueue domain collects global statistics for enqueue requests.

Enqueue global statistics

These statistics fields contain the global data collected by the enqueue domain for enqueue requests.

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS ENQUEUE command, and are mapped by the DFHNQGDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 42. Enqueue domain: enqueue requests global statistics

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	NQGPOOL	is the number of enqueue pools. Reset characteristic: not reset
ENQ Poolname	NQGNPOOL	is the enqueue pool id. Reset characteristic: not reset
ENQs Issued	NQGTNQSI	is the total number of enqueue requests issued. Reset characteristic: reset to zero
ENQs Waited	NQGTNQSW	is the total number of enqueue requests that had waited due to the enqueues being held. This is a subset of NQGTNQSI. Note that this value does not include the enqueue requests currently waiting (see NQGCNQSW). Reset characteristic: reset to zero

Table 42. Enqueue domain: enqueue requests global statistics (continued)

DFHSTUP name	Field name	Description
Enqueue Waiting time	NQGTNQWT	is the total waiting time for the enqueue requests that waited (NQGTNQSW). Note that this value does not include the time for the enqueue requests currently waiting (see NQGCNQWT). Reset characteristic: reset to zero
NOT IN THE DFHSTUP REPORT	NQGCNQSW	is the current number of enqueue requests waiting. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	NQGCNQWT	is the total waiting time for the enqueue requests that are currently waiting due to the enqueue being held by another transaction. Reset characteristic: not reset
Sysplex Waited	NQGGNQSW	is the total number of sysplex enqueue requests that had waited due to the enqueues being held.
Sysplex Waiting time	NQGGNQWT	Reset characteristic: reset to zero is the total waiting time for the sysplex enqueue requests that waited (NQGGNQSW). Reset characteristic: reset to zero
NOT IN THE DFHSTUP REPORT	NQGSNQSW	is the current number of sysplex enqueues waiting. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	NQGSNQWT	is the total waiting time for the sysplex enqueues that are currently waiting (NQGSNQSW). Reset characteristic: not reset
Enqueues Retained	NQGTNQSR	is the total number of enqueues that were retained due to the owning UOW being shunted. Note that this value does not include the enqueues that are currently retained (see NQGCNQSR). For more information about shunted UOWs see "Recovery manager" on page 439. Reset characteristic: reset to zero

Table 42. Enqueue domain: enqueue requests global statistics (continued)

DFHSTUP name	Field name	Description
Enqueue Retention time	NQGTNQRT	is the total retention time for the enqueues that were retained due to the owning UOW being shunted. Note that this value does not include the enqueue retention time for those currently retained (see NQGCNQRT). For more information about shunted UOWs see "Recovery manager" on page 439. Reset characteristic: reset to zero
NOT IN THE DFHSTUP REPORT	NQGCNQSR	is the current number of enqueues retained. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	NQGCNQRT	is the current enqueue retention time. Reset characteristic: not reset
Immediate-rejection		
–Enqbusy	NQGTIRJB	is the total number of enqueue requests that were immediately rejected due to the enqueue being busy (ENQBUSY response). This value is a subset of the total number of enqueue requests (NQGTNQSI). Reset characteristic: reset to zero
–Retained	NQGTIRJR	is the total number of enqueue requests that were immediately rejected due to the enqueue being in a retained state. This value is a subset of the total number of enqueue requests (NQGTNQSI). Reset characteristic: reset to zero
Waiting rejection		
–Retained	NQGTWRJR	is the total number of waiting enqueue requests that were rejected due to the required enqueue moving into a retained state. This value is a subset of the number of enqueue requests that waited (NQGTNQSW). Reset characteristic: reset to zero
–Operator	NQGTWPOP	is the total number of waiting enqueue requests that were rejected due to the operator purging the waiting transaction. This value is a subset of the number of enqueue requests that waited (NQGTNQSW). Reset characteristic: reset to zero
–Timeout	NQGTWPTO	is the total number of waiting enqueue requests that were rejected due to the timeout value (DTIMEOUT) being exceeded. This value is a subset of the number of enqueue requests that waited (NQGTNQSW). Reset characteristic: reset to zero

Enqueue: summary global statistics

Summary statistics are not available online.

These statistics fields contain the enqueue summary global data.

Table 43. Enqueue: Summary global statistics

DFHSTUP name	Description
ENQ Poolname	is the enqueue pool id.
ENQs Issued	is the total number of enqueue requests that were issued.
ENQs Waited	is the total number of enqueues requests that waited.
Enqueue Waiting time	is the waiting time for enqueue requests that waited.
Sysplex Waited	is the total number of sysplex enqueue requests that had waited due to the enqueues being held.
Sysplex Waiting time	is the total waiting time for the sysplex sysplex enqueue requests that waited.
Enqueues Retained	is the total number of enqueues retained.
Enqueue Retention time	is the enqueue retention time.
Immediate-rejection	
–Enqbusy	is the total number of enqueue requests that were immediately rejected ENQBUSY.
–Retained	is the total number of enqueue requests immediately rejected due to the enqueue being in a retained state.
Waiting rejection	
–Retained	is the total number of waiting enqueue requests that were rejected due to the required enqueue moving into a retained state.
–Operator	is the total number of waiting enqueue requests that were rejected due to the operator purging the waiting transaction.
–Timeout	is the total number of waiting enqueue requests that were rejected due to the timeout value being exceeded.

Front end programming interface (FEPI)

FEPI statistics contain data about the use of each FEPI connection, each FEPI pool, and a target in any pool

FEPI: Connection statistics

These statistics give information about each FEPI connection. The statistics are available online, and are mapped by the DFHA23DS DSECT.

Table 44. FEPI: connection statistics

DFHSTUP name	Field name	Description
Pool Name	A23POOL	is the FEPI pool name. Reset characteristic: not reset
Target Name	A23TARG	is the FEPI target name. Reset characteristic: not reset
Node Name	A23NODE	is the FEPI node. Reset characteristic: not reset
Acquires	A23ACQ	is the number of times the connection was acquired. Reset characteristic: reset to zero

Table 44. FEPI: connection statistics (continued)

DFHSTUP name	Field name	Description
Conversations	A23CNV	is the number of conversations that have used this connection. Reset characteristic: reset to zero
Unsolicited Inputs	A23USI	is the number of times unsolicited input was received on this connection. Reset characteristic: reset to zero
Characters		
–Sent	A23CHOUT	is the number of characters of data sent on this connection. Reset characteristic: reset to zero
–Received	A23CHIN	is the number of characters of data received on this connection. Reset characteristic: reset to zero
Receive Timeouts	A23RTOUT	is the number of times a FEPI RECEIVE timed-out on this connection. Reset characteristic: reset to zero
Error Conditions	A23ERROR	is the number of VTAM error conditions raised for this connection. Reset characteristic: reset to zero

FEPI: pool statistics

These statistics give information about each FEPI pool. The statistics are available online, and are mapped by the DFHA22DS DSECT.

Table 45. FEPI: Pool statistics

DFHSTUP name	Field name	Description
Pool Name	A22POOL	is the FEPI pool name. Reset characteristic: not reset
Targets	A22TRGCT	is the current number of targets in the pool. Reset characteristic: not reset
Nodes	A22NDCT	is the current number of nodes in the pool. Reset characteristic: not reset
Available Connections		
–Current	A22CONCT	is the number of connections in the pool. Reset characteristic: not reset
–Peak	A22CONPK	is the peak number of connections in the pool. This field is needed because targets and nodes may be deleted between intervals. Reset characteristic: reset to current value (A22CONCT)
Allocates		
–Total	A22ALLOC	is the number of conversations that have been allocated from this pool. Reset characteristic: reset to zero

Table 45. FEPI: Pool statistics (continued)

DFHSTUP name	Field name	Description
–Peak	A22PKALL	is the peak number of concurrent conversations allocated from this pool. Reset characteristic: reset to current value
Allocate Waits		
NOT IN THE DFHSTUP REPORT	A22WAIT	is the current number of conversations waiting to be allocated. Reset characteristic: not reset
–Total	A22TOTWT	is the number of conversations that had to wait to be allocated. Reset characteristic: reset to zero
–Peak	A22PKWT	is the peak number of conversations that had to wait to be allocated. Reset characteristic: reset to current value (A22WAIT)
Allocate Timeouts	A22TIOUT	is the number of conversation allocates that timed out. Reset characteristic: reset to zero

FEPI: target statistics

These statistics give information about a particular target in a pool. The statistics are available online, and are mapped by the DFHA24DS DSECT.

Table 46. FEPI: target Statistics

DFHSTUP name	Field name	Description
Target name	A24TARG	is the FEPI target name. Reset characteristic: not reset
Pool name	A24POOL	is the FEPI pool name. Reset characteristic: not reset
Applid	A24APPL	is the VTAM applid of the target. Reset characteristic: not reset
Nodes	A24NDCT	is the number of nodes connected to this target. Reset characteristic: not reset
Allocates	A24ALLOC	is the number of conversations **specifically** allocated to this target in this pool. Reset characteristic: reset to zero
Allocate Waits		
–Total	A24TOTWT	is the number of conversations that had to wait to be allocated to this target in this pool. Reset characteristic: reset to zero
–Wait	A24WAIT	is the number of current conversations waiting to be allocated to this target in this pool Reset characteristic: reset to zero

Table 46. FEPI: target Statistics (continued)

DFHSTUP name	Field name	Description
–Peak	A24PKWT	is the peak number of conversations that had to wait to be allocated to this target in this pool. Reset characteristic: reset to current value (A24WAIT)
Allocate Timeouts	A24TIOUT	is the number of conversation allocates to this target in this pool that timed out. Reset characteristic: reset to zero

FEPI: unsolicited connection statistics

Unsolicited connection statistics are produced when a connection is destroyed. This occurs when a DELETE POOL, DISCARD NODELIST, DISCARD POOL or DISCARD TARGETLIST command is used. The statistics are mapped by the DFHA23DS DSECT. They contain the same information as the interval statistics.

FEPI: unsolicited pool statistics

Unsolicited pool statistics are produced when a pool is discarded. The statistics are mapped by the DFHA22DS DSECT. They contain the same information as the interval statistics.

FEPI: unsolicited target statistics

Unsolicited target statistics are produced when a target is destroyed or removed from a pool. This occurs when a DELETE POOL, DISCARD POOL or DISCARD TARGETLIST command is used. The statistics are mapped by the DFHA24DS DSECT. They contain the same information as the interval statistics.

FEPI: connection summary statistics

Summary statistics are not available online.

Table 47. FEPI: connection summary statistics

DFHSTUP name	Description
Pool name	is the FEPI pool name.
Target name	is the FEPI target name.
Node name	is the FEPI node.
Acquires	is the total number of times the connection was acquired.
Conversations	is the total number of conversations that have used this connection.
Unsolicited Inputs	is the total number of times unsolicited input was received on this connection.
Characters Sent	
–Sent	is the total number of characters of data sent on this connection.
–Received	is the total number of characters of data received on this connection.
Receive timeouts	is the total number of times a FEPI RECEIVE timed-out on this connection.
Error conditions	is the total number of VTAM error conditions raised for this connection.

FEPI: pool summary statistics

Summary statistics are not available online.

Table 48. FEPI: pool summary statistics

DFHSTUP name	Description
Pool name	is the FEPI pool name.

Table 48. FEPI: pool summary statistics (continued)

DFHSTUP name	Description
Targets	is the number of targets in the pool.
Nodes	is the number of nodes in the pool.
Available connections	
–Current	is the number of connections in the pool.
–Peak	is the highest peak number of connections in the pool.
Allocates	
–Totals	is the total number of conversations allocated from this pool.
–Peak	is the highest peak number of concurrent conversations allocated from this pool.
Allocate waits	
–Total	is the total number of conversations that had to wait to be allocated.
–Peak	is the highest peak number of conversations that had to wait to be allocated.
Allocate timeouts	is the total number of conversation allocates that timed out.

FEPI: target summary statistics

Summary statistics are not available online.

Table 49. FEPI: target summary statistics

DFHSTUP name	Description
Target name	is the FEPI target name.
Pool name	is the FEPI pool name.
Applid	is the VTAM applid of the target.
Nodes	is the number of nodes in the pool.
Allocates	is the total number of conversations **specifically** allocated to this target in this pool.
Allocate waits	
–Total	is the total number of conversations that had to wait to be allocated to this target in this pool.
–Peak	is the highest peak number of conversations that had to wait to be allocated to this target in this pool.
Allocate timeouts	is the total number of conversations allocated to this target in this pool that timed out.

File control

There are four sections in the DFHSTUP report for file statistics:

- Files: resource information ("Files: resource information statistics" on page 380).
- Files: requests information ("Files: requests information statistics" on page 383).
- Files: data table requests information ("Files: data table requests information statistics" on page 385).
- Files: performance information ("Files: performance information statistics" on page 389).

Unsolicited file statistics are printed in a statistics report separate from other types of CICS statistics.

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS FILE command, and are mapped by the DFHA17DS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Files: resource information statistics

Table 50. Files: resource information statistics

DFHSTUP name	Field name	Description
File name	A17FNAM	is the name you specified in the DEFINE FILE command of resource definition online. Reset characteristic: not reset
Dataset name	A17DSNAM	is the 44-character name defining the physical data set to the system. You may have specified this in: • The DSNAME operand specified in the DEFINE FILE command of resource definition online • The operand specified in the DD DSN= operand of the CICS JCL • By dynamic allocation of a data set to a file through the use of CEMT SET FILE DSNAME or EXEC CICS SET FILE DSNAME commands. If no data set is currently allocated to the file, this field is blank. If the file is remote, no data set name is printed but the word "**remote**" is substituted for the data set name. Reset characteristic: not reset
Base dataset name (if applicable)	A17BDSNM	In the instance that the file is a VSAM PATH, this field gives the base data set name. Reset characteristic: not reset.
Dataset type	A17DSTYP	is the data set type. • B = BDAM • E = VSAM ESDS • K = VSAM KSDS • R = VSAM RRDS • P = VSAM PATH Reset characteristic: not reset.
RLS file	A17DSRLS	is an indicator of whether the file is RLS or not. • 'R' =RLS accessed file • ' ' =Non-RLS Reset characteristic: not reset.

Table 50. Files: resource information statistics (continued)

DFHSTUP name	Field name	Description
DataTable indicator	A17DT	is a one-byte field that contains the value "R", or "S" or "T", or "L" or "K", or "X", if data table statistics fields are present in the record. "R" indicates that this is a remote file for which table read and source read statistics are present. "S" indicates that the resource was not opened as a table but was able to access data from a table associated with the same data set. "T" indicates that the resource is a shared data table. "L" indicates that the resource is a coupling facility data table. "K" indicates that the resource is a coupling facility data table (contention model) "X" indicates that the resource has been opened with a source data set which has an associated CICS maintained data table.
Time opened	A17LOPNT	Reset characteristic: not reset is the time at which this file was opened. If this field is not set, A17LOPNT contains the hexadecimal value X'00000000 00000000', shown in the report as "CLOSED". If the field is set, it contains a time expressed as a store clock (STCK) value in local time. This field contains a valid time if: • The file was open at the time the statistics were taken. • This is an unsolicited statistics request due to the file being closed.
Time closed	A17LCLST	Reset characteristic: not reset is the time at which this file was closed. If this field is not set, A17LCLST contains the hexadecimal value X'00000000 00000000', shown in the report as "OPEN". If the field is set, it contains a time expressed as a store clock (STCK) value in local time.
Remote Name	A17RNAME	Reset characteristic: not reset The name by which this file is known in the system or region in which it is resident.
Remote Sysid	A17RSYS	Reset characteristic: not reset. When operating in an ISC or MRO environment, and the file is held by a remote system, this field specifies the system upon which the file is resident. Reset characteristic: not reset.

Table 50. Files: resource information statistics (continued)

DFHSTUP name	Field name	Description
LSR Pool ID	A17POOL	The identity of the local shared resource pool. This value is that specified by: • The LSRPOOLID operand of the resource definition online DEFINE FILE command. "N" means that it is not defined in an LSR pool. Reset characteristic: not reset.
CFDT PoolName	A17DTCFP	The name of the coupling facility data table pool defined for the data table associated with the file Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	A17FLOC	states whether the file is defined as being local to this CICS system, or resides on a remote CICS system. The field is one byte long, and is set to "R" if remote. Reset characteristic: not reset

Note: When the source data set of a user-maintained table is closed, the "time opened" is reset to the time at which the source was closed.

Files: summary resource information statistics

Summary statistics are not available online.

Table 51. Files: summary resource information statistics

DFHSTUP name	Description
File Name	is the name you specified in: • The DEFINE FILE command of resource definition online
Dataset name	is the 44-character name defining the physical data set to the system.
Base dataset name (If applicable)	In the instance that the file is a VSAM PATH, this field gives the base data set name.
Dataset type	is the data set type. • B = BDAM • E = VSAM ESDS • K = VSAM KSDS • R = VSAM RRDS • P = VSAM PATH
RLS file	is an indicator of whether the file is RLS accessed or not.
Data Table indicator	is a one-byte field that contains the value "R", or "S" or "T", or "X", if data table statistics fields are present in the record. "R" indicates that this is a remote file for which table read and source read statistics are present. "S" indicates that the resource was not opened as a table but was able to access data from a table associated with the same data set. "T" indicates that the resource is a data table. "X" indicates that the resource has been opened with a source data set which has an associated CICS maintained data table.
Remote name	The name by which this file is known in the system or region in which it is resident.

Table 51. Files: summary resource information statistics (continued)

DFHSTUP name	Description
Remote sysid	When operating in an ISC or MRO environment, and the file is held by a remote system, this field specifies the system upon which the file is resident.
LSR Pool ID	The identity of the local shared resource pool. This value is that specified via: • The LSRPOOLID operand of the resource definition online DEFINE FILE command. • The TYPE=FILE, LSRPOOL operand of the DFHFCT macro. "N" means that it is not defined in an LSR pool.
CFDT PoolName	The name of the coupling facility data table pool defined for the data table associated with the file.

Files: requests information statistics

The following eight items are service request statistics. They do not tell you directly how many I/O accesses are being carried out for each transaction (a single-transaction measurement is required for this). Nevertheless, by regularly totaling the service requests against individual data sets, they can enable you to anticipate data set problems when I/O activity increases.

They list the number of service requests processed against the data set. These are dependent on the type of requests that are allowed on the data set.

Table 52. Files: requests information statistics

DFHSTUP name	Field name	Description
File name	A17FNAM	is the name you specified in: • The DEFINE FILE command of resource definition online • The TYPE=FILE, FILE operand of the DFHFCT macro.
GET requests	A17DSRD	Reset characteristic: not reset is the number of GET requests attempted against this file.
GET upd requests	A17DSGU	Reset characteristic: reset to zero is the number of GET UPDATE requests attempted against this file.
Browse requests	A17DSBR	Reset characteristic: reset to zero is the number of GETNEXT and GETPREV requests attempted against this file.
Update requests	A17DSWRU	Reset characteristic: reset to zero is the number of PUT UPDATE requests attempted against this file.
Add requests	A17DSWRA	Reset characteristic: reset to zero is the number of PUT requests attempted against this file.
Delete requests		Reset characteristic: reset to zero A17DSDEL or A17RMDEL is printed here

Table 52. Files: requests information statistics (continued)

DFHSTUP name	Field name	Description
	A17DSDEL	is the number of DELETE requests attempted against this local file. Reset characteristic: reset to zero
	A17RMDEL	is the number of DELETE requests for a VSAM file in a remote system. In systems connected by a CICS intercommunication (MRO or ISC) link, the statistics recorded for the remote files are a subset of those recorded for the files on the local system. Reset characteristic: reset to zero
Brws upd requests	A17DSBRU	is the number of browse READNEXT and READPREV requests issued against this file. Note that this field is only applicable to RLS accessed files. Reset characteristic: reset to zero
VSAM EXCP requests		
–Data	A17DSXCP	A value is printed if the FCT entry has been opened and used as a VSAM KSDS during the CICS run, even if the FCT entry is not being used as a KSDS at the time of taking statistics. See notes 1, 2 and 3.
–Index	A17DSIXP	See notes 1, 2 and 3.
RLS req timeouts	A17RLSWT	Reset characteristic: reset to zero is the number of RLS requests made to this file that were not serviced in the specified time limit, and therefore the requests were terminated. Reset characteristic: reset to zero

Notes: The "VSAM EXCP requests" fields indicate the number of I/O operations on the file for data and index records respectively. Also, note the following points:

1. The values printed for both items relate to the FCT entry. If dynamic allocation has been used to change the physical data sets associated with an FCT entry, the value shown is an accumulation for all the data sets.
2. Take care when using these values for files participating in data set name sharing, because VSAM maintains only one count of EXCPs for all access method control blocks (ACBs) thus connected. In this situation, the value reported against each file represents the total accesses for all sharing ACBs during the period for which the file was open. (Therefore, if all files in the data set name sharing group were open for the same period, each file would have the same EXCP values reported against it, which would be the total for all the files in the group.)
3. For RLS, this value is a count of the number of calls to the system buffer manager. It includes calls that result in either a coupling facility cache access or an I/O.

Files: summary requests information statistics

Summary statistics are not available online.

Table 53. Files: summary requests information statistics

DFHSTUP name	Description
File name	is the name you specified in: • The DEFINE FILE command of resource definition online • The TYPE=FILE, FILE operand of the DFHFCT macro.
Get requests	is the total number of GET requests issued against this file.
Get upd requests	is the total number of GET UPDATE requests issued against this file.
Browse requests	is the total number of GETNEXT and GETPREV requests issued against this file.
Update requests	is the total number of PUT UPDATE requests issued against this file.
Add requests	is the total number of PUT requests issued against this file.
Delete requests	is the total number of DELETE requests issued against this file.
Brws upd requests	is the total number of READNEXT and READPREV requests issued against this file (RLS only).
VSAM EXCP request	
–Data	A value is printed if the FCT entry has been opened and used as a VSAM KSDS during the CICS run. See notes 1, 2 and 3.
–Index	See notes 1, 2 and 3.
RLS req timeouts	is the total number of RLS requests made to this file that were not serviced in the specified time limit, and therefore the requests were terminated.

Notes: The "VSAM EXCP requests" fields indicate the number of I/O operations on the file for data and index records respectively. Also, note the following points:

1. The values printed for both items relate to the FCT entry. If dynamic allocation has been used to change the physical data sets associated with an FCT entry, the value shown is an accumulation for all the data sets.
2. Take care when using these values for files participating in data set name sharing, because VSAM maintains only one count of EXCPs for all ACBs thus connected. In this situation, the value reported against each file represents the total accesses for all sharing ACBs during the period for which the file was open. (Therefore, if all files in the data set name sharing group were open for the same period, each file would have the same EXCP values reported against it, which would be the total for all the files in the group.)
3. For RLS, this value is a count of the number of calls to the system buffer manager. It includes calls that result in either a coupling facility cache access or an I/O.

Files: data table requests information statistics

If the file is a data table, further fields are present in the statistics record. The presence of these additional fields is indicated by the value "R", or "S", or "L", or "K", or "T", or "X" in the field A17DT. Their names and meanings are as follows:

Table 54. Files: data table requests statistics

DFHSTUP name	Field name	Description
File Name	A17FNAM	is the name you specified in: • The DEFINE FILE command of resource definition online • The TYPE=FILE, FILE operand of the DFHFCT macro. Reset characteristic: not reset

Table 54. Files: data table requests statistics (continued)

DFHSTUP name	Field name	Description
Close type	A17DTTYP	This one-byte field is set to: "C" when a CICS maintained table is closed "P" when a file which has been accessing a CICS-maintained table is closed but the table remains open because there are other files still open which are using the table "S" when the source data set for a user table is being closed "U" when a user maintained table is closed. "L" when a locking model coupling facility data table is closed "K" when a contention model coupling facility data table is closed.
Read requests	A17DTRDS	Reset characteristic: not reset is the number of attempts to retrieve records from the table.
Recs¬in table	A17DTRNF	Reset characteristic: reset to zero is the number of records placed in the table by the loading process or as a result of API READ requests issued while loading was in progress.
Adds from reads	A17DTAVR	Reset characteristic: reset to zero is the number of records placed in the table by the loading process or as a result of API READ requests issued while loading was in progress.
Add requests	A17DTADS	Reset characteristic: reset to zero is the number of attempts to add records to the table as a result of WRITE requests.
Adds rejected – exit	A17DTARJ	Reset characteristic: reset to zero is the number of records CICS attempted to add to the table which were rejected by the global user exit.
Adds rejected – table full	A17DTATF	Reset characteristic: reset to zero is the number of records CICS attempted to add to the table but was unable to do so because the table already contained the maximum number of records specified.
Rewrite requests	A17DTRWS	Reset characteristic: reset to zero is the number of attempts to update records in the table as a result of REWRITE requests.
Delete requests	A17DTDLS	Reset characteristic: reset to zero is the number of attempts to delete records from the table as a result of DELETE requests.
		Reset characteristic: reset to zero

Table 54. Files: data table requests statistics (continued)

DFHSTUP name	Field name	Description
Highest table size	A17DTSHI	is the peak number of records present in the table. Reset characteristic: reset at close
Storage alloc(K)	A17DTALT	is the total amount of storage allocated to the data table. The DFHSTUP report expresses the storage in kilobytes. DFHSTUP does not total the storage allocated for all data tables because multiple files may share the same data table. Reset characteristic: not reset
Chng Resp/Lock Waits	A17DTCON	For a CFDT that is using the locking model, records are locked down when they are read for update. This count is the number of times it was necessary to WAIT for an already locked record. For a CFDT that is using the contention model, records are not locked when they are read for update. If a subsequent rewrite or delete request finds that the record has already changed, a CHANGED response is returned. This count is the number of times that a CHANGED response was issued. Reset characteristic: reset to zero
NOT IN THE DFHSTUP REPORT	A17DTLDS	is the number of times that a LOADING response was issued. When a CFDT is in the process of being loaded, and requests issued for records beyond the range of those already loaded will get a LOADING response. Reset characteristic. reset to zero

Note: The request information statistics output for a data table represents the activity of the source data set, and the data table request information represents the activity of the data table. Thus, for a CICS-maintained table, you would expect to find similar counts in both sections of the statistics output for requests which modify the table, because both the source data set and the table must be updated. For a user-maintained table, there should be no updating activity shown in the data table resource information.

Table 55. Files: shared data table statistics

DFHSTUP name	Field name	Description
When using the shared data tables feature the statistics records will contain the additional information as follows:		
NOT IN THE DFHSTUP REPORT	A17DTSIZ	is the current number of records in the data table. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	A17DTUST	is the total amount of storage (kilobytes) in use for the data table. Reset characteristic: not reset

Table 55. Files: shared data table statistics (continued)

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	A17DTALE	is the total amount of storage (kilobytes) allocated for the record entry blocks. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	A17DTUSE	is the total amount of storage (kilobytes) in use for the record entry blocks. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	A17DTALI	is the total amount of storage (kilobytes) allocated for the index. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	A17DTUSI	is the total amount of storage (kilobytes) in use for the index. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	A17DTALD	is the total amount of storage (kilobytes) allocated for the record data. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	A17DTUSD	is the total amount of storage (kilobytes) in use for the record data. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	A17DTRRS	is the total number of read retries, that is the number of times reads in an AOR had to be retried because the FOR changed the table during the read. A17DTRRS is **not** a count of accesses which failed because a file owning region (FOR) was updating the specific record that the AOR wished to read. In such cases, the request is function shipped and is counted in the "source reads". Reset characteristic: not reset

Note: Data table fields are present in the statistics records but contain zeros if shared data tables are not installed or the resource is not a data table.

Files: summary data table requests statistics

Summary statistics are not available online.

Table 56. Files: summary data table requests statistics

DFHSTUP name	Description
File Name	is the name you specified in: • The DEFINE FILE command of resource definition online • The TYPE=FILE, FILE operand of the DFHFCT macro.
Table type	This one-byte field is set to: "C" when a CICS maintained table is closed. "P" when a file which has been accessing a CICS maintained table is closed but the table remains open because there are other files still open which are using the table, "S" when the source data set for a user table is being closed, "U" when a user maintained table is closed.

Table 56. Files: summary data table requests statistics (continued)

DFHSTUP name	Description
Successful reads	is the total number of reads from the data table.
Recs - in table	is the total number of records in the data table.
Adds from reads	is the total number of records placed in the table by the loading process or as a result of API READ requests issued while loading was in progress.
Add requests	is the total number of attempts to add records to the table as a result of WRITE requests.
Adds rejected	
–Exit	is the total number of records CICS attempted to add to the table which were rejected by the global user exit.
–Table full	is the total number of records CICS attempted to add to the table but was unable to do so because the table already contained the maximum number of records specified.
Rewrite requests	is the total number of attempts to update records in the table as a result of REWRITE requests.
Delete requests	is the total number of attempts to delete records from the table as a result of DELETE requests.
Highest table size	is the peak number of records present in the table.
Chng Resp/Lock Waits	is the total number of CHANGED responses encountered during the data table loading process.
Chng Resp/Lock Waits	is the total number of LOADING responses encountered during the data table loading process.

Files: performance information statistics

These statistics are available online, and are mapped by the DFHA17DS DSECT.

Table 57. Files: performance information statistics

DFHSTUP name	Field name	Description
File name	A17FNAM	is the name you specified in: • The DEFINE FILE command of resource definition online • The TYPE=FILE, FILE operand of the DFHFCT macro. Reset characteristic: not reset
Strings	A17STRNO	The maximum permissible number of concurrent updates. For RLS, the value specified in the ACB macro is ignored. After OPEN a value of 1024 is returned, indicating the maximum number of strings allowed. Reset characteristic: not reset.
Active strings	A17DSASC	The current number of updates against the file. Reset characteristic: not reset.
Wait on strings		
–Current	A17DSCWC	The current number of 'waits' for strings against the file. Reset characteristic: not reset
–Total	A17DSTSW	The total number of 'waits' for strings against the file. Reset characteristic: reset to zero

Table 57. Files: performance information statistics (continued)

DFHSTUP name	Field name	Description
–Highest	A17DSHSW	The highest number of 'waits' for strings against the file. Reset characteristic: reset to current value
Buffers		
–Data	A17DSDNB	The number of buffers to be used for data. For RLS, BUFND is ignored and the value specified in the ACB is returned. This parameter has no effect for hierarchical file systems (HFS) files. Reset characteristic: not reset.
–Index	A17DSINB	The number of buffers to be used for index. For RLS, BUFNI is ignored and the value specified in the ACB is returned. This parameter has no effect for hierarchical file systems (HFS) files. Reset characteristic: not reset.

Files: summary performance information statistics

Summary statistics are not available online.

Table 58. Files: summary performance information statistics

DFHSTUP name	Description
File name	is the name you specified in: • The DEFINE FILE command of resource definition online • The TYPE=FILE, FILE operand of the DFHFCT macro.
Strings	The maximum permissible number of concurrent updates. For RLS, the value specified in the ACB macro is ignored. After OPEN a value of 1024 is returned, indicating the maximum number of strings allowed.
Wait on strings	
–Total	The total number of ‘waits’ for strings against the file.
–HWM	The highest number of 'waits' for strings against the file.
Buffers	
–Data	The number of buffers to be used for data. For RLS, BUFND is ignored and the value specified in the ACB is returned. This parameter has no effect for hierarchical file systems (HFS) files.
–Index	The number of buffers to be used for index. For RLS, BUFNI is ignored and the value specified in the ACB is returned. This parameter has no effect for hierarchical file systems (HFS) files.

ISC/IRC system and mode entries

The ISC/IRC system and mode entry statistics area of the DFHSTUP listing is for a CICS system using intersystem communication. This provides summary statistics for the CICS intercommunication facility.

System entry

ISC/IRC system entry: Resource statistics

The system entry statistics record both ISC and IRC statistics. Some entries are unique to one or the other, and show zero activity if that function is not used. Statistics are provided for each system entry in the terminal definition.

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS CONNECTION command, and are mapped by the DFHA14DS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

This DSECT is to be used:

- For processing data returned for an online enquiry for a connection (EXEC CICS COLLECT STATISTICS)
- For processing connection statistics offline (SMF)
- For processing the connection totals (the summation of all defined connections in this CICS region).

CICS always allocates a SEND session when sending an IRC request to another region. Either a SEND or RECEIVE session can be allocated when sending requests using LU6.1 ISC, and either a contention loser or a contention winner session can be allocated when sending requests using APPC.

In LU6.1, SEND sessions are identified as secondaries, and RECEIVE sessions are identified as primaries.

Table 59. ISC/IRC system entry: resource statistics

DFHSTUP name	Field name	Description
Connection name	A14CNTN	corresponds to each system entry defined by a CONNECTION definition in the CSD, or by autoinstall. Reset characteristic: not reset
Connection netname	A14ESID	is the name by which the remote system is known in the network—that is, its applid. Reset characteristic: not reset
Access Method / Protocol	A14ACCM	is the communication access method used for this connection. The values are: • X'01' =A14VTAM • X'02' =A14IRC • X'03' =A14XM • X'04' =A14XCF
	A14EFLGS	is the communication protocol used for this connection. The values are: • X'01' =A14APPC • X'02' =A14LU61 • X'03' =A14EXCI Reset characteristic: not reset

Table 59. ISC/IRC system entry: resource statistics (continued)

DFHSTUP name	Field name	Description
Autoinstalled Connection Create Time	A14AICT	is the time at which this connection was autoinstalled, in local time. The time is expressed as *hours:minutes:seconds.decimals*. The DSECT field contains the value as a store clock (STCK). This field is only applicable to an autoinstalled APPC connection. For all other types of connection the value will be nulls (x'00').
Autoinstalled Connection Delete Time	A14AIDT	is the time at which this connection was deleted, in local time. The time is expressed as *hours:minutes:seconds.decimals*. The DSECT field contains the value as a store clock (STCK). This field is only set if this is an autoinstalled APPC connection that has been deleted, that is, this field is only set in an unsolicited statistics (USS) record. For all other types of connection and all other types of statistics record the value will be nulls (x'00').
Send session count	A14ESECN	is the number of SEND sessions for this connection. This field applies to MRO and LU6.1 connections only. Reset characteristic: not reset
Receive session count	A14EPRMN	is the number of RECEIVE sessions for this connection. This field applies to MRO and LU6.1 connections only. Reset characteristic: not reset
AIDs in chain	A14EALL	is the current number of automatic initiate descriptors (AIDs) in the AID chain. Reset characteristic: not reset
Generic AIDs in chain	A14ESALL	is the current number of automatic initiate descriptors (AIDs) that are waiting for a session to become available to satisfy an allocate request. Reset characteristic: not reset
ATIs satisfied by contention losers	A14ES1	is the number of ATI requests (queued allocates) that have been satisfied by contention loser sessions (primaries for LU6.1). This is always zero for IRC system entries. For APPC, this field is zero when written to SMF, but if accessed online using the EXEC CICS COLLECT STATISTICS command, this field is the summation of the equivalent mode entry statistics. Reset characteristic: reset to zero

Table 59. ISC/IRC system entry: resource statistics (continued)

DFHSTUP name	Field name	Description
ATIs satisfied by contention winners	A14ES2	is the number of ATI requests (queued allocates) that have been satisfied by contention winner sessions (secondaries for LU6.1). This field is the total ATIs when the system entry is for IRC. For APPC, this field is zero when written to SMF, but if accessed online using the EXEC CICS COLLECT STATISTICS command, this field is the summation of the equivalent mode entry statistics.
Current contention losers	A14E1RY	Reset characteristic: reset to zero is the number of contention loser sessions (primaries for LU6.1) that are currently in use.
Peak contention losers	A14E1HWM	Reset characteristic: not reset is the peak number of contention loser sessions (primaries for LU6.1) that were in use at any one time. For APPC, this field is zero.
Current contention winners	A14E2RY	Reset characteristic: reset to current value is the number of contention winner sessions (secondaries for LU6.1) that are currently in use.
Peak contention winners	A14E2HWM	Reset characteristic: not reset is the peak number of contention winner sessions (secondaries for LU6.1) that were in use at any one time. For APPC, this field is zero.
Total bids sent	A14ESBID	Reset characteristic: reset to current value is the total number of bids that were sent. A bid is sent on an LU6.1 RECEIVE session only. This field is always zero for IRC entries. For APPC, this field is zero when written to SMF, but if accessed online using the EXEC CICS COLLECT STATISTICS command, this field is the summation of the equivalent mode entry statistics.
Current bids in progress	A14EBID	Reset characteristic: reset to zero is the number of bids currently in progress. A bid is sent on an LU6.1 RECEIVE session only. This field is always zero for IRC system entries. For APPC, this field is zero when written to SMF, but if accessed online using the EXEC CICS COLLECT STATISTICS command, this field is the summation of the equivalent mode entry statistics. Reset characteristic: not reset

Table 59. ISC/IRC system entry: resource statistics (continued)

DFHSTUP name	Field name	Description
Peak bids in progress	A14EBHWM	is the peak number of bids that were in progress at any one time. A bid is sent on an LU6.1 RECEIVE session only. Reset characteristic: reset to current value
Note for the following five fields: For APPC only, if an allocate request does not specify a mode group, CICS takes the first mode group within the sessions available, and the statistics for these allocates are reported against the system entry. If an allocate specifically requests a mode entry, the statistics for these allocates go into that mode entry.		
Peak outstanding allocates	A14ESTAM	is the peak number of allocate requests that were queued for this system. For APPC this field is incremented only for generic allocate requests. Reset characteristic: reset to current value
Total number of allocates	A14ESTAS	is the number of allocate requests against this system. For APPC: • This field is incremented only for generic allocate requests • If accessed online using the EXEC CICS COLLECT STATISTICS command, this field also contains the summation of the equivalent mode entry statistics. Reset characteristic: reset to zero
Queued allocates	A14ESTAQ	is the current number of queued allocate requests against this system. An allocate is queued due to a session not being available at this moment. This includes waiting for a bind, a bid, or all sessions are currently in use. For APPC: • This field is incremented only for generic allocate requests • If accessed online using the EXEC CICS COLLECT STATISTICS command, this field also contains the summation of the equivalent mode entry statistics. Reset characteristic: not reset
Failed link allocates	A14ESTAF	is the number of allocate requests that failed due to the connection being released, out of service, or with a closed mode group. For APPC: • This field is incremented only for generic allocate requests • If accessed online using the EXEC CICS COLLECT STATISTICS command, this field also contains the summation of the equivalent mode entry statistics. Reset characteristic: reset to zero

Table 59. ISC/IRC system entry: resource statistics (continued)

DFHSTUP name	Field name	Description
Failed allocates due to sessions in use	A14ESTAO	is the number of allocate requests that failed due to a session not being currently available for use. These requests get SYSBUSY responses to the allocate. This field is incremented for allocates failing with an AAL1 abend code. For APPC only: • This field is only incremented for generic allocate requests • If accessed online using the EXEC CICS COLLECT STATISTICS command, this field also contains the summation of the equivalent mode entry statistics. Reset characteristic: reset to zero
Maximum queue time (seconds)	A14EMXQT	is the MAXQTIME specified on the CONNECTION definition. This value represents the maximum time you require to process an allocate queue on this connection. If the allocate queue would take greater than this time to process then the entire queue would be purged. This value only takes effect if the QUEUELIMIT value (A14EALIM) has been reached. Reset characteristic: not reset
Allocate queue limit	A14EALIM	is the QUEUELIMIT parameter specified on the CONNECTION definition. If this value is reached then allocates are rejected. Reset characteristic: not reset
Number of QUEUELIMIT allocates rejected	A14EALRJ	the total number of allocates rejected due to the QUEUELIMIT value (A14EALIM) being reached. Reset characteristic: reset to zero
Number of MAXQTIME allocate queue purges	A14EQPCT	is the total number of times an allocate queue has been purged due to the MAXQTIME value (A14EMXQT). A queue is purged when the total time it would take to process a queue exceeds the MAXQTIME value. Reset characteristic: reset to zero
Number of MAXQTIME allocates purged	A14EMQPC	is the total number of allocates purged due to the queue processing time exceeding the MAXQTIME value (A14EMXQT). If sessions have not been freed after this mechanism has been invoked then any subsequent allocate requests are purged and included in this statistic as the MAXQTIME purging mechanism is still in operation. Reset characteristic: reset to zero

Table 59. ISC/IRC system entry: resource statistics (continued)

DFHSTUP name	Field name	Description
Number of XZIQUE allocates rejected	A14EZQRJ	is the total number of allocates rejected by the XZIQUE exit.
Number of XZIQUE allocate queue purges	A14EZQPU	Reset characteristic: reset to zero is the total number of allocate queue purges that have occurred at XZIQUE request for this connection. If accessed online using the EXEC CICS COLLECT STATISTICS command, this field additionally contains the summation of the equivalent mode entry statistics.
Number of XZIQUE allocates purged	A14EZQPC	Reset characteristic: reset to zero is the total number of allocates purged due to XZIQUE requesting that queues should be purged (A14EZQPU) for this connection. If XZIQUE has not overridden this mechanism (by response) then any subsequent allocate requests are purged and included in this statistic as the XZIQUE purging mechanism is still in operation. If accessed online using the EXEC CICS COLLECT STATISTICS command, this field additionally contains the summation of the equivalent mode entry statistics.
File control (FC) function shipping requests	A14ESTFC	Reset characteristic: reset to zero is the number of file control requests for function shipping.
Interval control IC) function shipping requests	A14ESTIC	Reset characteristic: reset to zero is the number of interval control requests for function shipping.
Program control (PC) function shipping requests	A14ESTPC	Reset characteristic: reset to zero is the number of program control link requests for function shipping.
Transient data (TD) function shipping requests	A14ESTTD	Reset characteristic: reset to zero is the number of transient data requests for function shipping.
Temporary storage (TS) function shipping requests	A14ESTTS	Reset characteristic: reset to zero is the number of temporary storage requests for function shipping.
DL/I function shipping requests	A14ESTDL	Reset characteristic: reset to zero is the number of DL/I requests for function shipping.
		Reset characteristic: reset to zero

Table 59. ISC/IRC system entry: resource statistics (continued)

DFHSTUP name	Field name	Description
Terminal sharing requests	A14ESTTC	is the number of transaction routing commands. This number is incremented on both regions when the transaction is routed, and when the terminal I/O request is routed between regions. This field is not supported for LU6.1. Reset characteristic: reset to zero
NOT IN THE DFHSTUP REPORT	A14GACT	is the time at which this connection was autoinstalled, in GMT. The time is expressed as *hours:minutes:seconds.decimals*. The DSECT field contains the value as a store clock (STCK). This field is only applicable to an autoinstalled APPC connection. For all other types of connection the value will be nulls (x'00'). Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	A14GADT	is the time at which this connection was deleted, in GMT. The time is expressed as *hours:minutes:seconds.decimals*. The DSECT field contains the value as a store clock (STCK). This field is only set if this is an autoinstalled APPC connection that has been deleted, that is, this field is only set in an unsolicited statistics (USS) record. For all other types of connection and all other types of statistics record the value will be nulls (x'00'). Reset characteristic: not reset

ISC/IRC system entry: Summary resource statistics

Summary statistics are not available online.

Table 60. ISC/IRC system entry: Summary resource statistics

DFHSTUP name	Description
Connection name	is the system entry defined by the CONNECTION definition in the CSD or by autoinstall.
Connection netname	is the name by which the remote system is known in the network—that is, its applid.
Access Method / Protocol	is the combined communication access method and protocol used for the connection.
Average autoinstalled connection time	is the average autoinstalled connection time. This field applies to autoinstalled connections and is summarized from the unsolicited system entry statistics records only.
Send session count	is the last value encountered for the SENDCOUNT specified on the CONNECTION definition. This field applies to MRO and LU6.1 connections only.
Receive session count	is the last value encountered for the RECEIVECOUNT specified on the CONNECTION definition. This field applies to MRO, LU6.1, and EXCI connections only.
Average number of AIDs in chain	is the average number of automatic initiate descriptors (AIDs) in the AID chain.

Table 60. ISC/IRC system entry: Summary resource statistics (continued)

DFHSTUP name	Description
Average number of generic AIDs in chain	is the average number of AIDs waiting for a session to become available to satisfy an allocate request.
ATIs satisfied by contention losers	is the total number of ATI requests (queued allocates) that have been satisfied by contention loser sessions (primaries for LU6.1). This is always zero for IRC system entries.
ATIs satisfied by contention winners	is the total number of ATI requests (queued allocates) that have been satisfied by contention winner sessions (secondaries for LU6.1). This field is the total ATIs when the system entry is for IRC.
Peak contention losers	is the peak number of contention loser sessions (primaries for LU6.1). that were in use at any one time. For APPC, this field is zero.
Peak contention winners	is the peak number of contention winner sessions (secondaries for LU6.1) that were in use at any one time. For APPC, this field is zero.
Total bids sent	is the total number of bids that were sent. A bid is sent on an LU6.1 RECEIVE session only. This field is always zero for IRC and APPC system entries.
Average bids in progress	is the average number of bids in progress. A bid is sent on an LU6.1 RECEIVE session only. This field is always zero for IRC and APPC system entries.
Peak bids in progress	is the peak number of bids that were in progress at any one time. A bid is sent on an LU6.1 RECEIVE session only. This field is always zero for IRC and APPC system entries.
Note for the following five fields: For APPC only, if an allocate request does not specify a mode group, CICS takes the first mode group within the sessions available, and the statistics for these allocates are reported against the system entry. If an allocate specifically requests a mode entry, the statistics for these allocates go into that mode entry.	
Peak outstanding allocates	is the peak number of allocation requests that were queued for this system. For APPC this field contains only generic allocate requests.
Total number of allocates	is the total number of allocate requests against this system. For APPC this field contains only generic allocate requests.
Average number of queued allocates	is the average number of queued allocate requests against this system. For APPC this field is incremented only for generic allocate requests.
Failed link allocates	is the total number of allocate requests that failed due to the connection being released, out of service, or with a closed mode group. For APPC this field is incremented only for generic allocate requests.
Failed allocates due to sessions in use	is the total number of allocate requests that failed due to a session not being currently available for use. These requests get SYSBUSY responses to the allocate. This field is incremented for allocates failing with an AAL1 abend code. For APPC this field is incremented only for generic allocate requests.
Maximum queue time (seconds)	is the last non-zero value encountered for the MAXQTIME specified on the CONNECTION definition. This value represents the maximum time you require to process an allocate queue on this connection. If the allocate queue would take greater than this time to process the entire queue would be purged. This value only takes effect if the QUEUELIMIT value has been reached.
Allocate queue limit	is the last non-zero value encountered for the QUEUELIMIT parameter specified on the CONNECTION definition. If this value is reached then allocates are rejected.
Number of QUEUELIMIT allocates rejected	is the is the total number of allocates rejected due to the QUEUELIMIT value being reached.

Table 60. ISC/IRC system entry: Summary resource statistics (continued)

DFHSTUP name	Description
Number of MAXQTIME allocate queue purges	is the total number of times an allocate queue has been purged due to the MAXQTIME value. A queue is purged when the total time it would take to process a queue exceeds the MAXQTIME value.
Number of MAXQTIME allocates purged	is the total number of allocates purged due to the queue processing time exceeding the MAXQTIME value. If sessions have not been freed after this mechanism has been invoked then any subsequent allocate requests are purged and included in this statistic as the MAXQTIME purging mechanism is still in operation.
Number of XZIQUE allocates rejected	is the total number of allocates rejected by the XZIQUE exit
Number of XZIQUE allocate queue purges	is the total number of allocate queue purges that have occurred at XZIQUE request for this connection.
Number of XZIQUE allocates purged	is the total number of allocates purged due to XZIQUE requesting that queues should be purged for this connection. If XZIQUE has not overridden this mechanism (by response) then any subsequent allocate requests are purged and included in this statistic as the XZIQUE purging mechanism is still in operation.
File control (FC) function shipping requests	is the total number of file control requests for function shipping.
Interval control (IC) function shipping requests	is the total number of interval control requests for function shipping.
Program control (PC) function shipping requests	is the total number of program control link requests for function shipping.
Transient data (TD) function shipping requests	is the total number of transient data requests for function shipping.
Temporary storage (TS) function shipping requests	is the total number of temporary storage requests for function shipping.
DL/I function shipping requests	is the total number of DL/I requests for function shipping.
Terminal sharing requests	is the total number of transaction routing commands. This number is incremented on both regions when the transaction is routed, and when the terminal I/O request is routed between regions. This field is not supported for LU6.1.

Mode entry

ISC mode entry: Resource statistics

These statistics are collected only if you have an APPC connection defined in your CICS region, and they are then produced for each mode group defined in that connection. These statistics cannot be accessed online using the EXEC CICS COLLECT STATISTICS command. They are only produced for offline processing (written to SMF).

These statistics are mapped by the DFHA20DS DSECT. This DSECT is also used to map the mode entry totals records.

Table 61. ISC/IRC mode entry: Resource statistics

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	A20SYSN	is the name of the APPC connection/system that owns this mode entry. It corresponds to the system entry, defined by a CONNECTION definition in the CSD or by autoinstall.
Mode name	A20MODE	Reset characteristic: not reset is the mode group name related to the the intersystem connection name above (A20SYSN). This corresponds to modename in the sessions definition.
ATIs satisfied by contention losers	A20ES1	Reset characteristic: not reset is the number of ATI requests (queued allocates) that have been satisfied by "contention loser" sessions belonging to this mode group.
ATIs satisfied by contention winners	A20ES2	Reset characteristic: reset to zero is the number of ATI requests (queued allocates) that have been satisfied by "contention winner" sessions belonging to this mode group.
Current contention losers	A20E1RY	Reset characteristic: reset to zero is the current number of "contention loser" sessions.
Peak contention losers	A20E1HWM	Reset characteristic: not reset is the peak number of "contention loser" sessions belonging to this mode group that were in use at any one time. There can be sessions not defined (by the MAXIMUM parameter of CEDA) as "contention winners" or "contention losers", and their states are dynamically decided at bind time.
Current contention winners	A20E2RY	Reset characteristic: reset to current value is the current number of "contention winner" sessions.
Peak contention winners	A20E2HWM	Reset characteristic: not reset is the peak number of "contention winner" sessions belonging to this mode group that were in use at any one time. There can be sessions not defined (by the MAXIMUM= parameter of CEDA) as "contention winners" or "contention losers", and their states are dynamically decided at bind time.
Total bids sent	A20ESBID	Reset characteristic: reset to current value is the number of bids that were sent on the sessions defined to this mode group. A bid is sent on an APPC "contention loser" session when there are no "contention winner" sessions available to allocate. Reset characteristic: reset to zero

Table 61. ISC/IRC mode entry: Resource statistics (continued)

DFHSTUP name	Field name	Description
Current bids in progress	A20EBID	is the number of bids that are in progress on the sessions defined to this mode group. A bid is sent on an APPC "contention loser" session when there are no "contention winner" sessions available to allocate. Reset characteristic: not reset
Peak bids in progress	A20EBHWM	is the peak number of bids that were in progress at any one time, on the sessions defined to this mode group. A bid is sent on an APPC "contention loser" session when there are no "contention winner" sessions available to allocate. Reset characteristic: reset to current value
The following three fields are incremented when an allocate is issued against a specific mode group. If a generic allocate request is made, the equivalent system entry statistics ***only*** are incremented.		
Peak outstanding allocates	A20ESTAM	is the peak number of allocation requests that were queued for this mode group. Reset characteristic: reset to current value
Total specific allocate requests	A20ESTAS	is the number of specific allocate requests against this mode group. Reset characteristic: reset to zero
Total specific allocates satisfied	A20ESTAP	is the number of specific allocates satisfied by this mode group. Reset characteristic: reset to zero
Total generic allocates satisfied	A20ESTAG	is the number of generic allocates satisfied from this mode group. The allocates are made for APPC without the mode group being specified. Reset characteristic: reset to zero
The following three fields are incremented when an allocate is issued against a specific mode group. If a generic allocate request is made, the equivalent system entry statistics ***only*** are incremented.		
Queued allocates	A20ESTAQ	is the current number of queued specific allocate requests against this mode group. An allocate is queued due to a session in this mode group not being available at this moment. This includes waiting for a bind, a bid, or all sessions are currently in use. Reset characteristic: not reset
Failed link allocates	A20ESTAF	is the number of specific allocate requests that failed due to the connection being released, out of service, or with a closed mode group. Reset characteristic: reset to zero

Table 61. ISC/IRC mode entry: Resource statistics (continued)

DFHSTUP name	Field name	Description
Failed allocates due to sessions in use	A20ESTAO	is the number of specific allocate requests that failed due to a session not being currently available for use in this mode group. These requests get SYSBUSY responses to the allocate. This field is incremented for allocates failing with an AAL1 abend code. Reset characteristic: reset to zero
Number of XZIQUE allocate queue purges	A20EQPCT	is the total number of allocate queue purges that have occurred at XZIQUE request for this mode entry. Reset characteristic: reset to zero
Number of XZIQUE allocates purged	A20EZQPC	is the total number of allocates purged due to XZIQUE requesting that queues should be purged (A20EQPCT) for this mode entry. If XZIQUE has not overridden this mechanism (by response) then any subsequent allocate requests are purged and included in this statistic as the XZIQUE purging mechanism is still in operation. Reset characteristic: reset to zero
Maximum session count	A20ELMAX	is the maximum number of sessions that the definition of the session group permits. Reset characteristic: not reset
Current maximum session count	A20EMAXS	is the current number of sessions in the group (the number "bound"). Reset characteristic: not reset
Maximum contention winners acceptable	A20EMCON	is the maximum number of sessions that the definition of the session group permits to be contention winners. Reset characteristic: not reset
Current contention losers in use	A20ECONL	is the number of contention loser sessions that are currently in use. Reset characteristic: not reset
Current contention winners in use	A20ECONW	is the number of contention winner sessions that are currently in use. Reset characteristic: not reset

ISC mode entry: Summary resource statistics

Summary statistics are not available online.

These statistics are collected only if you have an APPC connection defined in your CICS region, and they are then produced for each mode group defined in that connection.

Table 62. ISC/IRC mode entry: Summary resource statistics

DFHSTUP name	Description
Connection name	is the name of the APPC connection/system that owns this mode entry. It corresponds to the system entry in the terminal definition.
Mode name	is the mode group name related to the intersystem connection name above (A20SYSN). It corresponds to the modename in the sessions definition.
ATIs satisfied by contention losers	is the total number of ATI requests (queued allocates) that have been satisfied by "contention loser" sessions belonging to this mode group.
ATIs satisfied by contention winners	is the total number of ATI requests (queued allocates) that have been satisfied by "contention winner" sessions belonging to this mode group.
Peak contention losers	is the peak number of "contention loser" sessions belonging to this mode group that were in use at any one time. There can be sessions not defined (by the MAXIMUM= parameter of CEDA) as "contention winners" or "contention losers", and their states are dynamically decided at bind time.
Peak contention winners	is the peak number of "contention winner" sessions belonging to this mode group that were in use at any one time. There can be sessions not defined (by the MAXIMUM= parameter of CEDA) as "contention winners" or "contention losers", and their states are dynamically decided at bind time.
Total bids sent	is the total number of bids that were sent on the sessions defined to this mode group. A bid is sent on an APPC "contention loser" session when there are no "contention winner" sessions available to allocate.
Average bids in progress	is the average number of bids in progress.
Peak bids in progress	is the peak number of bids that were in progress at any one time, on the sessions defined to this mode group. A bid is sent on an APPC "contention loser" session when there are no "contention winner" sessions available to allocate.
The next three fields only contain allocates against specific mode groups. Generic allocate requests are contained in the equivalent system entry statistics.	
Peak outstanding allocates	is the peak number of allocation requests that were queued for this mode group.
Total specific allocate requests	is the total number of specific allocate requests against this mode group.
Total specific allocates satisfied	is the total number of specific allocates satisfied by this mode group.
Total generic allocates satisfied	is the total number of generic allocates satisfied from this mode group.
The next three fields only contain allocates against specific mode groups. Generic allocate requests are contained in the equivalent system entry statistics.	
Average number of queued allocates	is the average number of queued specific allocate requests against this mode group. An allocate is queued due to a session in this mode group not being available at this moment. This includes waiting for a bind, a bid, or all sessions are currently in use.
Failed link allocates	is the total number of specific allocate requests that failed due to the connection being released, out of service, or with a closed mode group.
Failed allocates due to sessions in use	is the total number of specific allocate requests that failed due to a session not being currently available for use in this mode group. These requests get SYSBUSY responses to the allocate. This field is incremented for allocates failing with an AAL1 abend code.
Number of XZIQUE allocate queue purges	is the total number of allocate queue purges that have occurred at XZIQUE request for this mode entry.

Table 62. ISC/IRC mode entry: Summary resource statistics (continued)

DFHSTUP name	Description
Number of XZIQUE allocates purged	is the total number of allocates purged due to XZIQUE requesting that queues should be purged (A20EQPCT) for this mode entry. If XZIQUE has not overridden this mechanism (by response) then any subsequent allocate requests are purged and included in this statistic as the XZIQUE purging mechanism is still in operation.

ISC/IRC attach time entries

The ISC/IRC attach time statistics of the DFHSTUP listing is for a CICS system using intersystem communication or interregion communication. It provides summary statistics for the number of times that the entries on the PV 'signed on from' list are either reused or timed out. Using this data you can adjust the USRDELAY, and the PVDELAY system initialization parameters.

ISC/IRC attach time: resource statistics

These statistics are collected if you have either an LU6.2 connection or IRC defined in your CICS region, and they are then produced globally, one per system. These statistics cannot be accessed online using the EXEC CICS COLLECT STATISTICS command; they are only produced for offline processing (written to SMF).

These statistics are mapped by the DFHA21DS DSECT.

Table 63. ISC/IRC attach time: resource statistics

DFHSTUP name	Field name	Description
Persistent Verification refresh time	A21_SIT_LUIT_TIME	is the time in minutes set by the PVDELAY system initialization parameter. It specifies the password re-verification interval. The range is from zero through 10080 minutes (seven days) and the default is 30 minutes. If a value of zero is specified, entries are deleted immediately after use. Reset characteristic: not reset
ISC Persistent Verification Activity		
Entries reused	A21_LUIT_TOTAL_REUSES	refers to the number of entries in the PV 'signed on from' list of a remote system that were reused without reference to an external security manager (ESM), such as RACF. Reset characteristic: reset to zero
Entries timed out	A21_LUIT_TOTAL_TIMEOUT	refers to the number of entries in the PV 'signed on from' list of a remote system that were timed out. Reset characteristic: reset to zero

Table 63. ISC/IRC attach time: resource statistics (continued)

DFHSTUP name	Field name	Description
Average reuse time between entries	A21_LUIT_AV_REUSE_TIME	refers to the average time that has elapsed between each reuse of an entry in the PV 'signed on from' list of a remote system. Reset characteristic: reset to zero

ISC/IRC attach time: Summary resource statistics

Summary statistics are not available online.

These statistics are collected only if you have either an LU6.2 connection or IRC defined in your CICS region, and they are then produced globally, one per system.

Table 64. ISC/IRC attach time: Summary resource statistics

DFHSTUP name	Description
Persistent verification refresh time	is the time in minutes set by the PVDELAY parameter of the SIT. It specifies how long entries are allowed to remain unused in the PV 'signed on from' list of a remote system.
Entries reused	refers to the number of times that user's entries in the PV 'signed on from' list were reused without referencing the ESM of the remote system.
Entries timed out	refers to the number of user's entries in the PV 'signed on from' list that were timed out after a period of inactivity.
Average reuse time between entries	refers to the average amount of time that has elapsed between each reuse of a user's entry in the PV 'signed on from' list.

Journalname

Journalname resource statistics

These statistics fields contain the resource data collected by the log manager domain. For more information on logging and journaling, see "Chapter 23. Logging and journaling" on page 269, and "Journalname and log stream statistics" on page 50.

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS JOURNALNAME command, and are mapped by the DFHLGRDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 65. Journalname: resource statistics

DFHSTUP name	Field name	Description
Journal Name	LGRJNLNAME	is the journal name. Reset characteristic: not reset
Journal Type	LGRJTYPE	is the type of journal: MVS, SMF, or dummy. Reset characteristic: not reset

Table 65. Journalname: resource statistics (continued)

DFHSTUP name	Field name	Description
Log Stream Name	LGRSTREAM	is the log stream name associated with the journal. Only journals defined as type MVS have associated log streams. The same log stream can be associated with more than one journal. Reset characteristic: not reset
Write Requests	LGRWRITES	is the total number of times that a journal record was written to the journal. Reset characteristic: reset to zero
Bytes Written	LGRBYTES	is the total number of bytes written to the journal. Reset characteristic: reset to zero
Buffer Flushes	LGRBUFLSH	is the total number of times that a journal block was written to the log stream (in the case of a journal defined as type MVS), or to the System Management Facility (in the case of a journal defined as type SMF). Journal blocks are flushed in the following circumstances: • An application executes an EXEC CICS WRITE JOURNALNAME (or JOURNALNUM) command with the WAIT option. • An application executes an EXEC CICS WAIT JOURNALNAME (or JOURNALNUM) command. • The journal buffer is full. This applies only to journals defined as type SMF (journals defined as type MVS use log stream buffers). • The log stream buffer is full. This applies only to journals defined as type MVS. Reset characteristic: reset to zero

Journalname: Summary resource statistics

Summary statistics are not available online.

These statistics fields contain the journalname summary resource data.

Table 66. Journalname: Summary resource statistics

DFHSTUP name	Description
Journal Name	is the journal name.
Journal Type	is the journal type: • X'01' - MVS • X'02' - SMF • X'03' - dummy
Log Stream Name	is the name of the log stream associated with the journal.
Write Requests	is the total number of times that a journal record was written to the journal.
Bytes Written	is the total number of bytes written.

Table 66. Journalname: Summary resource statistics (continued)

DFHSTUP name	Description
Buffer Flushes	is the total number of times that a journal block was written to the log stream (in the case of a journal defined as type MVS), or to the System Management Facility (in the case of a journal defined as type SMF).

Log stream

Log stream resource statistics

These statistics fields contain the resource data collected by the log manager domain. For more information on logging and journaling, see "Chapter 23. Logging and journaling" on page 269, and "Journalname and log stream statistics" on page 50.

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS STREAMNAME command and are mapped by the DFHLGSDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 67. Log stream: resource statistics

DFHSTUP name	Field name	Description
Log Stream Name	LGSTRNAM	The log stream name.
System Log	LGSSYSLG	Reset characteristic: not reset Indicates if the log stream forms part of the System Log.
Structure Name	LGSSTRUC	Reset characteristic: not reset The coupling facility (CF) structure name for the log stream. The structure name is only applicable to coupling facility type logstreams.
Max Block Length	LGSMAXBL	Reset characteristic: not reset The maximum block size allowed by the MVS Logger for the log stream.
DASD Only	LGSDONLY	Reset characteristic: not reset Indicates the type of log stream. If set to 'YES' the log stream is of type DASDONLY. If set to 'NO' the log stream is of type coupling facility (CF).
Retention Period	LGSRETPD	Reset characteristic: not reset The log stream retention period (in days) that the data must be kept before it can be physically deleted by the MVS Logger.
		Reset characteristic: not reset

Table 67. Log stream: resource statistics (continued)

DFHSTUP name	Field name	Description
Auto Delete	LGSAUTOD	The log data auto delete indicator. If set to 'YES' the MVS Logger automatically deletes the data as it matures beyond the retention period, irrespective of any logstream delete calls. If set to 'NO' the data is only deleted when a logstream delete call is issued and the data has matured beyond the retention period. Reset characteristic: not reset
Delete Requests	LGSDELETES	The number of DELETEs of blocks of data from the log stream. Reset characteristic: reset to zero

Log stream request statistics

These statistics fields contain the request data collected by the log manager domain.

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS STREAMNAME command and are mapped by the DFHLGSDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 68. Log stream: request statistics

DFHSTUP name	Field name	Description
Log Stream Name	LGSTRNAM	The log stream name. Reset characteristic: not reset
Write Requests	LGSWRITES	The number of WRITEs of blocks of data to the log stream. Reset characteristic: reset to zero
Bytes Written	LGSBYTES	The total number of bytes written to the log stream Reset characteristic: reset to zero
Waits Buff Full	LGSBUFWAIT	The total number of attempts made to append a journal record to the current log stream buffer while the buffers were logically full. This situation arises when the current log stream buffer has insufficient space to accommodate the journal record, and I/O is already in progress for the alternate log stream buffer. Reset characteristic: reset to zero
Buffer Appends	LGSBUFAPP	The number of occasions on which a journal record was successfully appended to the current log stream buffer. Reset characteristic: reset to zero
Current Frce Wtrs	LGSCUFWTRS	The current number of tasks suspended whilst requesting a flush of the log stream buffer currently in use. Reset characteristic: not reset

Table 68. Log stream: request statistics (continued)

DFHSTUP name	Field name	Description
Peak Frce Wtrs	LGSPKFWTRS	The peak number of tasks suspended whilst requesting a flush of the log stream buffer currently in use. Reset characteristic: reset to current
Total Force Wts	LGSTFCWAIT	The total number of tasks suspending whilst requesting a flush of the log stream buffer currently in use. Reset characteristic: reset to zero
Browse Starts	LGSBRWSTRT	The number of BROWSE operations started on the log stream. Reset characteristic: reset to zero
Browse Reads	LGSBRWREAD	The number of READs of blocks of data from the log stream. Reset characteristic: reset to zero
Retry Errors	LGSRTYERRS	The number of occasions on which MVS system logger retryable errors occurred when a block of data was being written to the log stream. Reset characteristic: reset to zero

Log stream: Summary resource statistics

Summary statistics are not available online.

These statistics fields contain the log stream summary resource data.

Table 69. Log stream: Summary resource statistics

DFHSTUP name	Description
Log Stream Name	The log stream name.
System Log	Indicates if the log stream forms part of the System Log.
Stucture Name	The coupling facility (CF) structure name for the log stream. The structure name is only applicable to coupling facility type logstreams.
Max Block Length	The maximum block size allowed by the MVS Logger for the log stream.
DASD Only	Indicates the type of log stream. If set to 'YES' the log stream is of type DASDONLY. If set to 'NO' the log stream is of type coupling facility (CF).
Retention Period	The log stream retention period (in days) that the data must be kept before it can be physically deleted by the MVS Logger.
Auto Delete	The log data auto delete indicator. If set to 'YES' the MVS Logger automatically deletes the data as it matures beyond the retention period, irrespective of any logstream delete calls. If set to 'NO' the data is only deleted when a logstream delete call is issued and the data has matured beyond the retention period.
Log Delete Requests	The total number of DELETEs of blocks of data from the log stream.

Log stream: Summary request statistics

Summary statistics are not available online.

These statistics fields contain the log stream summary request data.

Table 70. Log stream: Summary request statistics

DFHSTUP name	Description
Log Stream Name	The log stream name.
Write Requests	The total number of WRITEs of blocks of data to the log stream.
Bytes Written	The total number of bytes written to the log stream.
Waits Buffer Full	The total number of attempts made to append a journal record to the current log stream while the buffers were logically full.
Buffer Appends	The total number of occasions on which a journal record was successfully appended to the current log stream buffer.
Current Force Wtrs	The current number of tasks suspended whilst requesting a FLUSH of the log stream buffer currently in use.
Peak Force Wtrs	The highest peak number of tasks suspended whilst requesting a FLUSH of the log stream buffer currently in use.
Total Force Waits	The total number of tasks suspended whilst requesting a FLUSH of the log stream buffer currently in use.
Log Browse Starts	The total number of BROWSE operations started on the log stream.
Log Browse Reads	The total number of READs of blocks of data from the log stream.
Retry Errors	The total number of occasions on which MVS system logger retryable errors occurred when a block of data was being written to the log stream.

LSRpool

CICS supports the use of up to eight LSRpools, and produces two sets of statistics for LSRpool activity.

LSRpool: resource statistics for each LSR pool

The following information describes the size and characteristics of the pool, and shows the data collected for the use of strings and buffers.

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS LSRPOOL command, and are mapped by the DFHA08DS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 71. LSRpool: resource statistics for each LSR pool

DFHSTUP name	Field name	Description
Pool Number	A08SRPID	is the identifying number of the pool. This value may be in the range 1 through 8. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	A08FLAGS	is a flag set to value X'80' if separate data and index pools are used, or set to value X'00' if data and index buffers share the same pool. Reset characteristic: not reset
Time Created	A08LKCTD	is the time when this LSR pool was created. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals* in local time. Reset characteristic: not reset

Table 71. LSRpool: resource statistics for each LSR pool (continued)

DFHSTUP name	Field name	Description
Time Deleted	A08LKDTD	is the local time (STCK) when this LSR pool was deleted. This field is printed only if the pool has been deleted (that is, if all the files using the pool have been closed). If no value is set, the DSECT field contains the packed hexadecimal value X'00000000 00000000'. This field is only printed for unsolicited statistics when the pool is deleted. The process of deleting an LSR pool results in the output of unsolicited statistics for the pool. Information for the deleted pool is not printed in subsequent statistics output. For this reason, the "time pool deleted" field is normally printed only in this unsolicited statistics output. Reset characteristic: not reset
NOT IN DFHSTUP REPORT	A08GBKCD	is the time when this LSR pool was created. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals* in GMT. Reset characteristic: not reset
NOT IN DFHSTUP REPORT	A08GBKDD	is the time when this LSR pool was deleted expressed in GMT. This field is printed only if the pool has been deleted (that is, if all the files using the pool have been closed). If no value is set, the DSECT field contains the packed hexadecimal value X'00000000 00000000' This field is only printed for unsolicited statistics when the pool is deleted. The process of deleting an LSR pool results in the output of unsolicited statistics for the pool. Information for the deleted pool is not printed in subsequent statistics output. For this reason, the "time pool deleted" field is normally printed only in this unsolicited statistics output. Reset characteristic: not reset
Maximum key length	A08BKKYL	is the length of the largest key of a VSAM data set which may use the LSR pool. The value is obtained from one of: • The MAXKEYLENGTH option of the DEFINE LSRPOOL command in resource definition online, if it has been coded • A CICS calculation at the time the LSR pool is built. Reset characteristic: not reset

Table 71. LSRpool: resource statistics for each LSR pool (continued)

DFHSTUP name	Field name	Description
Total number of strings	A08BKSTN	is the value obtained from one of: • The STRINGS option of the DEFINE LSR command in resource definition online, if it has been coded • A CICS calculation at the time the LSR pool is built. Reset characteristic: not reset
Peak requests that waited for string	A08BKHSW	is the highest number of requests that were queued at one time because all the strings in the pool were in use. Reset characteristic: reset to current value
Total requests that waited for string	A08BKTSW	is the number of requests that were queued because all the strings in the pool were in use. This number reflects the number of requests that were delayed during CICS execution due to a restriction in LSR pool string resources. Reset characteristic: reset to zero
Peak concurrently active strings	A08BKHAS	is the maximum number of strings that were active during CICS execution. If you have coded a value for the number of strings the pool is to use, this statistic is always less than or equal to the value you have coded. If your coded value for string numbers is consistently lower than this value in the statistics, you could consider reducing it so that your pool of VSAM strings is not bigger than you need. Reset characteristic: reset to current value

Note that if separate data and index pools are not being used, all the statistics for the totals are obtained from the A08TO*xxx*_DATA variables, the index totals being unused.

LSRpool: data buffer statistics

Table 72. LSRPOOL: data buffer statistics

DFHSTUP name	Field name	Description
Size	A08BKBSZ	is the size of the buffers that are available to CICS. Buffers may be specified through: • The DEFINE LSRPOOL command of resource definition online • A CICS calculation at the time the LSRPOOL is built, of the buffers to use. Reset characteristic: not reset
Number	A08TOBFN_DATA	is the number of data buffers used by the pool. Reset characteristic: not reset
Lookasides	A08TOBFF_DATA	is the number of successful lookasides to data buffers for the pool. Reset characteristic: not reset

Table 72. LSRPOOL: data buffer statistics (continued)

DFHSTUP name	Field name	Description
Reads	A08TOFRD_DATA	is the number of read I/Os to the data buffers for the pool. Reset characteristic: not reset
User writes	A08TOUIW_DATA	is the number of user-initiated buffer WRITEs from data buffers for the pool. Reset characteristic: not reset
Non-user writes	A08TONUW_DATA	is the number of non-user-initiated buffer WRITEs from data buffers for the pool. Reset characteristic: not reset

Hiperspace: data buffer statistics

Table 73. Hiperspace: data buffer statistics

DFHSTUP name	Field name	Description
Size	A08BKBSZ	is the size of the buffers that are available to CICS. Buffers may be specified through: • The DEFINE LSRPOOL command of resource definition online • A CICS calculation at the time the LSRPOOL is built, of the buffers to use. Reset characteristic: not reset
Number	A08TOHBN_DATA	is the number of Hiperspace data buffers specified for the pool Reset characteristic: not reset
Hiperspace reads	A08TOCRS_DATA	is the number of successful CREAD requests issued to transfer data from Hiperspace data buffers to virtual data buffers. Reset characteristic: not reset
Hiperspace writes	A08TOWRS_DATA	is the number of successful CWRITE requests issued to transfer data from virtual data buffers to Hiperspace data buffers. Reset characteristic: not reset
Hiperspace failed reads	A08TOCRF_DATA	is the number of CREAD requests that failed. MVS had withdrawn the space and VSAM had to read data from DASD. Reset characteristic: not reset
Hiperspace failed writes	A08TOCWF_DATA	is the number of CWRITE requests that failed. There was insufficient Hiperspace and VSAM had to write data to DASD. Reset characteristic: not reset

LSRpool: index buffer statistics

Table 74. LSRpool: index buffer statistics

DFHSTUP name	Field name	Description
Size	A08BKBSZ	is the size of the buffers that are available to CICS. Buffers may be specified through: • The DEFINE LSRPOOL command of resource definition online • A CICS calculation at the time the LSRPOOL is built, of the buffers to use. Reset characteristic: not reset
Number	A08TOBFN_INDX	is the number of index buffers used by the pool. Reset characteristic: not reset
Lookasides	A08TOBFF_INDX	is the number of successful lookasides to index buffers for the pool. Reset characteristic: not reset
Reads	A08TOFRD_INDX	is the number of read I/Os to the index buffers for the pool. Reset characteristic: not reset
User writes	A08TOUIW_INDX	is the number of user-initiated buffer WRITEs from index buffers for the pool. Reset characteristic: not reset
Non-user writes	A08TONUW_INDX	is the number of non-user-initiated buffer WRITEs from index buffers for the pool. Reset characteristic: not reset

Hiperspace index buffer statistics

Table 75. Hiperspace: index buffer statistics

DFHSTUP name	Field name	Description
Size	A08BKBSZ	is the size of the buffers that are available to CICS. Buffers may be specified through: • The DEFINE LSRPOOL command of resource definition online • A CICS calculation at the time the LSRPOOL is built, of the buffers to use. Reset characteristic: not reset
Number	A08TOHBN_INDX	is the number of Hiperspace index buffers specified for the pool Reset characteristic: not reset
Hiperspace reads	A08TOCRS_INDX	is the number of successful CREAD requests issued to transfer data from Hiperspace index buffers to virtual index buffers. Reset characteristic: not reset
Hiperspace writes	A08TOWRS_INDX	is the number of successful CWRITE requests issued to transfer data from virtual index buffers to Hiperspace index buffers. Reset characteristic: not reset

Table 75. Hiperspace: index buffer statistics (continued)

DFHSTUP name	Field name	Description
Hiperspace failed reads	A08TOCRF_INDX	is the number of CREAD requests that failed. MVS had withdrawn the space and VSAM had to read data from DASD. Reset characteristic: not reset
Hiperspace failed writes	A08TOCWF_INDX	is the number of CWRITE requests that failed. There was insufficient Hiperspace and VSAM had to write data to DASD. Reset characteristic: not reset

The following group of statistics fields describes the characteristics and usage of the different buffer sizes available for use by the pool. These statistics are available online, and are mapped by the A08BSSDS DSECT defined in the DFHA08DS DSECT. This DSECT is repeated for each of the 11 CISIZEs available.

LSRpool: Buffer statistics

Table 76. LSRpool: Buffer statistics

DFHSTUP name	Field name	Description
Buffer Size	A08BKBSZ	is the size of the buffers that are available to CICS. Buffers may be specified through: • The DEFINE LSRPOOL command of resource definition online • A CICS calculation at the time the LSRPOOL is built buffers to use. Reset characteristic: not reset
Number	A08BKBFN	lists the number of buffers of each size available to CICS: Reset characteristic: not reset
Lookasides	A08BKBFF	is the number of read requests that VSAM was able to satisfy without initiating an I/O operation; that is, the requested record, whether index or data, was already present in one of the buffer resident CIs. This means that no physical I/O had to be done to put the control interval in the buffer. The tuning methodology usually employed involves either increasing the number of buffers of a particular CI size until the ratio of lookasides to READs stops increasing significantly or, conversely, reducing the number of buffers until the ratio of lookasides to READs begins to drop significantly. For most data sets, successful lookaside hits on indexes are more likely. These statistics are obtained from VSAM and represent the activity after the pool was created. Note that these statistics are ***not*** reset by CICS under any circumstances. Reset characteristic: not reset

Table 76. LSRpool: Buffer statistics (continued)

DFHSTUP name	Field name	Description
Reads	A08BKFRD	is the number of I/O operations to the buffers that VSAM was required to initiate to satisfy the CICS application's activity. This figure represents failures to find the control interval in the buffers. These statistics are obtained from VSAM and represent the activity after the pool was created. Note that these statistics are ***not*** reset by CICS under any circumstances. Reset characteristic: not reset
User writes	A08BKUIW	is the number of user-initiated I/O WRITE operations from the buffers that VSAM was required to initiate to satisfy the CICS application's activity. These statistics are obtained from VSAM and represent the activity after the pool was created. Note that these statistics are ***not*** reset by CICS under any circumstances. Reset characteristic: not reset
Non-user writes	A08BKNUW	is the number of non-user initiated I/O WRITE operations from the buffers that VSAM was forced to initiate due to no buffers being available for reading the contents of a CI. These statistics are obtained from VSAM and represent the activity after the pool was created. Note that these statistics are ***not*** reset by CICS under any circumstances. Reset characteristic: not reset

Hiperspace buffer statistics

Table 77. Hiperspace: buffer statistics

DFHSTUP name	Field name	Description
Size	A08BKBSZ	is the size of the buffers that are available to CICS. Buffers may be specified through: • The DEFINE LSRPOOL command of resource definition online • A CICS calculation at the time the LSRPOOL is built, of the buffers to use. Reset characteristic: not reset
Number	A08BKHBN	is the number of Hiperspace buffers specified for the pool. Reset characteristic: not reset
Hiperspace reads	A08BKCRS	is the number of successful CREAD requests issued to transfer data from Hiperspace buffers to virtual buffers. Reset characteristic: not reset

Table 77. Hiperspace: buffer statistics (continued)

DFHSTUP name	Field name	Description
Hiperspace writes	A08BKCWS	is the number of successful CWRITE requests issued to transfer data from virtual buffers to Hiperspace buffers. Reset characteristic: not reset
Hiperspace failed reads	A08BKCRF	is the number of CREAD requests that failed. MVS had withdrawn the space and VSAM had to read data from DASD. Reset characteristic: not reset
Hiperspace failed writes	A08BKCWF	is the number of CWRITE requests that failed. There was insufficient Hiperspace and VSAM had to write data to DASD. Reset characteristic: not reset

These Hiperspace statistics are obtained from VSAM and represent the activity after the pool was created. Note that these statistics are ***not*** reset by CICS under any circumstances.

LSRpool: Summary resource statistics for each LSR pool

Summary statistics are not available online.

Table 78. LSRpool: Summary resource statistics for each LSR pool

DFHSTUP name	Description
Total number of pools built	is the total number of LSRPOOLS that were built during the entire CICS run.
Peak requests that waited for string	is the highest number of requests that were queued at one time because all the strings in the pool were in use.
Total requests that waited for string	is the total number of requests that were queued because all the strings in the pool were in use. This number reflects the number of requests that were delayed during CICS execution due to a restriction in LSR pool string resources.
Peak concurrently active strings	is the peak number of strings that were active during CICS execution. If you have coded a value for the number of strings the pool is to use, this statistic is always less than or equal to the value you have coded. If your coded value for string numbers is consistently lower than this value in the statistics, you could consider reducing it so that your pool of VSAM strings is not bigger than you need.

LSRpool: Summary data buffer statistics

Summary statistics are not available online.

The group of statistics fields below summarizes the usage of each of the eight LSRPOOLS during the entire CICS run.

Table 79. LSRpool: summary data buffer statistics

DFHSTUP name	Description
Pool Number	is the identifying number of the pool. This value may be in the range 1 through 8.
Lookasides	is the total number of successful lookasides to data buffers for the pool.
Reads	is the total number of read I/Os to the data buffers for the pool.
User writes	is the total number of user-initiated buffer WRITEs from data buffers for the pool.

Table 79. LSRpool: summary data buffer statistics (continued)

DFHSTUP name	Description
Non-user writes	is the total number of non-user-initiated buffer WRITEs from data buffers for the pool.

Summary Hiperspace data buffer statistics

Summary statistics are not available online.

Table 80. LSRpool: summary Hiperspace data buffer statistics

DFHSTUP name	Description
Pool Number	is the identifying number of the pool. This value may be in the range 1 through 8.
Hiperspace reads	is the total number of Hiperspace data buffers specified for the pool.
Hiperspace writes	is the total number of successful CWRITE requests issued to transfer data from virtual data buffers to Hiperspace data buffers.
Hiperspace failed reads	is the total number of CREAD requests that failed. MVS had withdrawn the space and VSAM had to read data from DASD.
Hiperspace failed writes	is the total number of CWRITE requests that failed. There was insufficient Hiperspace and VSAM had to write data to DASD.

LSRpool: summary index buffer statistics

Summary statistics are not available online.

Table 81. LSRpool: summary Hiperspace data buffer statistics

DFHSTUP name	Description
Pool Number	is the identifying number of the pool. This value may be in the range 1 through 8.
Lookasides	is the total number of successful lookasides to index buffers for the pool.
Reads	is the total number of read I/Os to the index buffers for the pool.
User writes	is the total number of user-initiated buffer WRITEs from index buffers for the pool.
Non-user writes	is the total number of non-user-initiated buffer WRITEs from index buffers for the pool.

LSRpool: Summary Hiperspace index buffer statistics

Summary statistics are not available online.

Table 82. LSRpool: Resource statistics for each LSR pool

DFHSTUP name	Description
Pool Number	is the identifying number of the pool. This value may be in the range 1 through 8.
Hiperspace reads	is the total number of successful CREAD requests issued to transfer data from Hiperspace index buffers to virtual index buffers.
Hiperspace writes	is the total number of successful CWRITE requests issued to transfer data from virtual index buffers to Hiperspace index buffers.
Hiperspace failed reads	is the total number of CREAD requests that failed. MVS had withdrawn the space and VSAM had to read data from DASD.
Hiperspace failed writes	is the total number of CWRITE requests that failed. There was insufficient Hiperspace and VSAM had to write data to DASD.

If LSRpool buffers are shared, the statistics that follow refer to those shared data and index buffers.

LSRpool: Summary resource statistics for each LSR pool

Summary statistics are not available online.

Table 83. LSRpool: Summary resource statistics for each LSR pool

DFHSTUP name	Description
Pool Number	is the identifying number of the pool. This value may be in the range 1 through 8.
Lookasides	is the total number of read requests that VSAM was able to satisfy without initiating an I/O operation; that is, the requested record, whether index or data, was already present in one of the buffer resident CIs. This means that no physical I/O had to be done to put the control interval in the buffer. The tuning methodology usually employed involves either increasing the number of buffers of a particular CI size until the ratio of lookasides to READs stops increasing significantly or, conversely, reducing the number of buffers until the ratio of lookasides to READs begins to drop significantly. For most data sets, successful lookaside hits on indexes are more likely. These statistics are obtained from VSAM and represent the activity after the pool was created. Note that these statistics are ***not*** reset by CICS under any circumstances.
Reads	is the total number of I/O operations to the buffers that VSAM was required to initiate to satisfy the CICS application's activity. This figure represents failures to find the control interval in the buffers. These statistics are obtained from VSAM and represent the activity after the pool was created. Note that these statistics are ***not*** reset by CICS under any circumstances.
User writes	is the total number of user-initiated I/O WRITE operations from the buffers that VSAM was required to initiate to satisfy the CICS application's activity. These statistics are obtained from VSAM and represent the activity after the pool was created. Note that these statistics are ***not*** reset by CICS under any circumstances.
Non-user writes	is the total number of non-user initiated I/O WRITE operations from the buffers that VSAM was forced to initiate due to no buffers being available for reading the contents of a CI. These statistics are obtained from VSAM and represent the activity after the pool was created. Note that these statistics are ***not*** reset by CICS under any circumstances.

LSRpool: summary Hiperspace buffer statistics

Summary statistics are not available online.

Table 84. LSRpool: summary Hiperspace buffer statistics

DFHSTUP name	Description
Pool Number	is the identifying number of the pool. This value may be in the range 1 through 8.
Hiperspace reads	is the total number of successful CREAD requests issued to transfer data from Hiperspace buffers to virtual buffers.
Hiperspace writes	is the total number of successful CWRITE requests issued to transfer data from virtual buffers to Hiperspace buffers.
Hiperspace failed reads	is the total number of CREAD requests that failed. MVS had withdrawn the space and VSAM had to read data from DASD.

Table 84. LSRpool: summary Hiperspace buffer statistics (continued)

DFHSTUP name	Description
Hiperspace failed writes	is the total number of CWRITE requests that failed. There was insufficient Hiperspace and VSAM had to write data to DASD. The above Hiperspace statistics are obtained from VSAM and represent the activity after the pool was created. Note that these statistics are ***not*** reset by CICS under any circumstances.

The following information describes the buffer usage for each file that was specified to use the LSR pool at the time the statistics were printed. Note that this section is not printed for unsolicited statistics output.

If the allocation of files to the LSR pool is changed during the period that the statistics cover, no history of this is available and only the current list of files sharing the pool are printed in this section. The activity of all files that have used the pool are, however, included in all the preceding sections of these statistics.

LSRpool Files: Resource statistics for each file specified to use the pool

Table 85. LSRpool Files: Resource statistics for each file specified

DFHSTUP name	Field name	Description
Pool Number	A09SRPID	is the LSR pool number, in the range 1 through 8, associated with this file. Reset characteristic: not reset
File Name	A09DSID	is the CICS file identifier you specified through resource definition online. Reset characteristic: not reset
Data Buff Size	A09DBN	is the buffer size used for the file's data records. This value is one of the eleven possible VSAM buffer sizes ranging from 512 bytes to 32 KB. The value is zero if the file has not been opened yet. Reset characteristic: not reset
Index Buff Size	A09IBN	is the buffer size used for the file's index records. This is printed, even if the file has subsequently been dynamically allocated to a VSAM ESDS or RRDS. The values this field may take are the same as for the data buffer size statistic. Reset characteristic: not reset
Total Buff Waits	A09TBW	is the number of requests that had to wait because all buffers of the size used by the data set for data (or index) in the LSR pool were in use. Reset characteristic: reset to zero

Table 85. LSRpool Files: Resource statistics for each file specified (continued)

DFHSTUP name	Field name	Description
Peak Buff Waits	A09HBW	is the peak number of requests that had to wait because all buffers of the size used by the data set for data (or index) in the LSR pool were in use. If the data sets are waiting for buffers you should examine the numbers of buffers defined for the data and index buffer sizes used by the data set. The buffer size used by VSAM depends on the control interval size in the VSAM definition of the data set. If no buffer size exists for the specified control interval size, the next largest buffer size available is used. Reset characteristic: reset to current value

LSRpool files: Summary resource statistics

Summary statistics are not available online.

Table 86. LSRpool files: Summary resource statistics

DFHSTUP name	Description
Pool Number	is the LSR pool number, in the range 1 through 8, associated with this file.
File Name	is the CICS file identifier you specified through resource definition online.
Data Buff Size	is the last non-zero value encountered for the buffer size used for the file's data records. This value is one of the eleven possible VSAM buffer sizes ranging from 512 bytes to 32 KB. The value is zero if the file has not been opened yet. The last non-zero value is produced only if it has been opened.
Index Buff Size	is the last non-zero value encountered for the buffer size used for the file's index records. This is printed, even if the file has subsequently been dynamically allocated to a VSAM ESDS or RRDS. The values this field may take are the same as for the data buffer size statistic.
Total Buff Waits	is the total number of requests that had to wait because all buffers of the size used by the data set for data (or index) in the LSR pool were in use.
Peak Buff Waits	is the peak number of requests that had to wait because all buffers of the size used by the data set for data (or index) in the LSR pool were in use. If the data sets are waiting for buffers you should examine the numbers of buffers defined for the data and index buffer sizes used by the data set. The buffer size used by VSAM depends on the control interval size in the VSAM definition of the data set. If no buffer size exists for the specified control interval size, the next largest buffer size available is used.

Monitoring domain

Monitoring data is made up of a combination of performance class data, exception class data, and SYSEVENT data.

Monitoring domain: global statistics

These statistics fields are collected from the monitoring domain. They can be accessed online using the EXEC CICS COLLECT STATISTICS MONITOR command, and are mapped by the DFHMNGDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 87. Monitoring domain: global statistics

DFHSTUP name	Field name	Description
Exception records	MNGER	is the number of exception records written to SMF. Reset characteristic: reset to zero
Exception records suppressed	MNGERS	is the number of exception records suppressed by the global user exit (XMNOUT). Reset characteristic: reset to zero
Performance records	MNGPR	is the number of performance records scheduled for output to SMF. Because the monitoring domain buffers performance class records, there is still a possibility that monitoring can be deactivated, thus preventing the output of the performance class records that have been buffered. Reset characteristic: reset to zero
Performance records suppressed	MNGPRS	is the number of performance records suppressed by the global user exit (XMNOUT). Reset characteristic: reset to zero
SMF records	MNGSMFR	is the number of SMF records written to the SMF data set. CICS writes exception class SMF records as soon as the monitor domain is notified of the exception completion, so there is one exception record per SMF record. The performance class, however, has many performance class records per SMF record. The SMF record for the performance class is written when the buffer is full, performance class has been deactivated, or CICS is quiescing. Reset characteristic: reset to zero
SMF errors	MNGSMFE	is the number of non-OK responses from the request to write a record to SMF. This count is incremented when an SMF write fails for any reason, for example, when SMF is inactive. Reset characteristic: reset to zero

Table 87. Monitoring domain: global statistics (continued)

DFHSTUP name	Field name	Description
Sysevent records	MNGSYSER	is the number of SYSEVENT notification records written to the MVS SRM (for later processing by RMF). A sysevent record is written at the completion of a transaction, or at suspend of a semi-permanent mirror (MRO only). Reset characteristic: reset to zero
Sysevent errors	MNGSYSEE	is the number of non-OK responses from the request to write a record to the MVS SRM (for later processing by RMF). This count is incremented when a SYSEVENT write fails for any reason, for example, when RMF in inactive. Reset characteristic: reset to zero

Monitoring domain: summary global statistics

Summary statistics are not available online.

Table 88. Monitoring domain: summary global statistics

DFHSTUP name	Description
Exception records	is the total number of exception records written to SMF.
Exception records suppressed	is the total number of exception records suppressed by the global user exit (XMNOUT).
Performance records	is the total number of performance records scheduled for output to SMF. Because the monitor domain buffers performance class records, there is still a possibility that monitoring can be deactivated, thus preventing the output of the performance class records that have been buffered.
Performance records suppressed	is the total number of performance records suppressed by the global user exit (XMNOUT).
SMF records	is the total number of SMF records written to the SMF data set. CICS writes exception class SMF records as soon as the monitor domain is notified of the exception completion, so there is one exception record per SMF record. The performance class, however, has many performance class records per SMF record. The SMF record for the performance class is written when the buffer is full, performance class has been deactivated, or CICS is quiescing.
SMF errors	is the total number of non-OK responses from the request to write a record to SMF. This count is incremented when an SMF write fails for any reason, for example, when SMF is inactive.
Sysevent records	is the total number of SYSEVENT notification records written to the MVS SRM (for later processing by RMF). A sysevent record is written at the completion of a transaction, or at suspend of a semi-permanent mirror (MRO only).
Sysevent errors	is the total number of non-OK responses from the request to write a record to the MVS SRM (for later processing by RMF). This count is incremented when a SYSEVENT write fails for any reason, for example, when RMF in inactive.

Program autoinstall

Program autoinstall: global statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS PROGAUTO command, and are mapped by the DFHPGGDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 89. Program autoinstall: global statistics

DFHSTUP name	Field name	Description
Program autoinstall attempts	PGGATT	is the number of times that a program autoinstall was attempted. Reset characteristic: reset to zero
Rejected by autoinstall exit	PGGREJ	is the number of times that a program autoinstall request was rejected by the program autoinstall URM program. Reset characteristic: reset to zero
Failed autoinstall attempts	PGGFAIL	is the number of times that a program autoinstall failed due to a number of reasons other than rejects (as counted by PGGREJ). For example the autoinstall URM program did not provide valid attributes; the model name specified by the URM was not defined; the exit tried to recurse, and disabled the URM. Reset characteristic: reset to zero

Program autoinstall: summary global statistics

Summary statistics are not available online.

Table 90. Program autoinstall: summary global statistics

DFHSTUP name	Description
Program autoinstall attempts	is the number of times that a program was autoinstalled.
Rejected by autoinstall exit	is the number of times that a program is rejected by the autoinstall exit.
Failed autoinstall attempts	is the number of times that a program failed to autoinstall.

Loader

Loader domain: global statistics

These statistics fields contain the global data collected by the loader domain. The loader domain maintains global statistics to assist the user in tuning and accounting.

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS PROGRAM command, and are mapped by the DFHLDGDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 91. Loader domain: global statistics

DFHSTUP name	Field name	Description
Library load requests	LDGLLR	is the number of times the loader has issued an MVS LOAD request to load programs from the DFHRPL library concatenation into CICS managed storage. Modules in the LPA are not included in this figure. Reset characteristic: reset to zero
Total loading time	LDGLLT	is the time taken for the number of library loads indicated by LDGLLR. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains a 4-byte field which expresses the time in 16-microsecond units. Reset characteristic: reset to zero
Average loading time		is the average time to load a program. This value is calculated offline by DFHSTUP and hence is not available to online users. DFHSTUP expresses this time as *hours:minutes:seconds.decimals*. Reset characteristic: none
Program uses	LDGPUSES	is the number of uses of any program by the CICS system. Reset characteristic: not reset
Waiting requests	LDGWLR	is the number of loader domain requests that ***are currently*** forced to suspend due to the loader domain currently performing an operation on that program on behalf of another task. These operations could be: • A NEWCOPY request • Searching the LPA • A physical load in progress. Reset characteristic: not reset
Requests that waited	LDGWTDLR	is the number of loader domain requests that ***were*** forced to suspend due to the loader domain performing an operation on that program on behalf of another task. These operations could be: • A NEWCOPY request • Searching the LPA • A physical load in progress. This figure is the total number of tasks that have waited, and does not include those that are currently waiting (LDGWLR). Reset characteristic: reset to zero
Peak waiting Loader requests	LDGWLRHW	is the maximum number of tasks suspended at one time. Reset characteristic: reset to current value (LDGWLR)

Table 91. Loader domain: global statistics (continued)

DFHSTUP name	Field name	Description
Times at peak	LDGHWMT	is the number of times the high watermark level indicated by LDGWLRHW was reached. This, along with the fields; LDGWTDLR and LDGWLRHW, is an indication of the level of contention for loader resource. Reset characteristic: reset to 1
Total waiting time	LDGTTW	is the suspended time for the number of tasks indicated by LDGWTDLR. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains a 4-byte field which expresses the time in 16-microsecond units. Reset characteristic: reset to zero
Times DFHRPL re-opened	LDGDREBS	is the number of times the loader received an end-of-extent condition during a LOAD and successfully closed and re-opened the DFHRPL library and retried the LOAD. Reset characteristic: reset to zero
CDSA		
Programs removed by compression	LDGDPSCR	is the number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism. Reset characteristic: reset to zero
Total Not In Use queue membership time	LDGDPSCT	is the program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed time CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value. Reset characteristic: reset to zero
Average Not In Use queue membership time		is the average length of time that a program is eligible for removal from storage by the DPSC mechanism. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value. Reset characteristic: none

Table 91. Loader domain: global statistics (continued)

DFHSTUP name	Field name	Description
Reclaims from Not In Use queue	LDGRNIU	is the number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC). Reset characteristic: reset to zero
Programs loaded but Not In Use	LDGPNIU	is the number of programs on the Not-In-Use (NIU) queue. Reset characteristic: not reset
Amount of DSA occupied by Not In Use programs	LDGCNIU	is the current amount of CDSA storage which is occupied by Not-In-Use (NIU) programs. Reset characteristic: not reset
ECDSA		
Programs removed by compression	LDGDPSCR	is the number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism. Reset characteristic: reset to zero
Total Not In Use queue membership time	LDGDPSCT	is the program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed time CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value.
Average Not In Use queue membership time		Reset characteristic: reset to zero is the average length of time that a program is eligible for removal from storage by the DPSC mechanism. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value. Reset characteristic: none

Table 91. Loader domain: global statistics (continued)

DFHSTUP name	Field name	Description
Reclaims from Not In Use queue	LDGRNIU	is the number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC). Reset characteristic: reset to zero
Programs loaded but Not In Use	LDGPNIU	is the number of programs on the Not-In-Use (NIU) queue. Reset characteristic: not reset
Amount of DSA occupied by Not In Use programs	LDGCNIU	is the current amount of ECDSA storage which is occupied by Not-In-Use (NIU) programs. Reset characteristic: not reset
SDSA		
Programs removed by compression	LDGDPSCR	is the number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism. Reset characteristic: reset to zero
Total Not In Use queue membership time	LDGDPSCT	is the program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed time CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value. Reset characteristic: reset to zero
Average Not In Use queue membership time		is the average length of time that a program is eligible for removal from storage by the DPSC mechanism. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value. Reset characteristic: none

Table 91. Loader domain: global statistics (continued)

DFHSTUP name	Field name	Description
Reclaims from Not In Use queue	LDGRNIU	is the number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC). Reset characteristic: reset to zero
Programs loaded but Not In Use	LDGPNIU	is the number of programs on the Not-In-Use (NIU) queue. Reset characteristic: not reset
Amount of DSA occupied by Not In Use programs	LDGCNIU	is the current amount of SDSA storage which is occupied by Not-In-Use (NIU) programs. Reset characteristic: not reset
ESDSA		
Programs removed by compression	LDGDPSCR	is the number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism. Reset characteristic: reset to zero
Total Not In Use queue membership time	LDGDPSCT	is the program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed time CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value. Reset characteristic: reset to zero
Average Not In Use queue membership time		is the average length of time that a program is eligible for removal from storage by the DPSC mechanism. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value. Reset characteristic: none

Table 91. Loader domain: global statistics (continued)

DFHSTUP name	Field name	Description
Reclaims from Not In Use queue	LDGRNIU	is the number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC). Reset characteristic: reset to zero
Programs loaded but Not In Use	LDGPNIU	is the number of programs on the Not-In-Use (NIU) queue. Reset characteristic: not reset
Amount of DSA occupied by Not In Use programs	LDGCNIU	is the current amount of ESDSA storage which is occupied by Not-In-Use (NIU) programs. Reset characteristic: not reset
RDSA		
Programs removed by compression	LDGDPSCR	is the number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism. Reset characteristic: reset to zero
Total Not In Use queue membership time	LDGDPSCT	is the program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed time CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value. Reset characteristic: reset to zero
Average Not In Use queue membership time		is the average length of time that a program is eligible for removal from storage by the DPSC mechanism. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value. Reset characteristic: none

Table 91. Loader domain: global statistics (continued)

DFHSTUP name	Field name	Description
Reclaims from Not In Use queue	LDGRNIU	is the number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC). Reset characteristic: reset to zero
Programs loaded but Not In Use	LDGPNIU	is the number of programs on the Not-In-Use (NIU) queue. Reset characteristic: not reset
Amount of DSA occupied by Not In Use programs	LDGCNIU	is the current amount of RDSA storage which is occupied by Not-In-Use (NIU) programs. Reset characteristic: not reset
ERDSA		
Programs removed by compression	LDGDPSCR	is the number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism. Reset characteristic: reset to zero
Total Not In Use queue membership time	LDGDPSCT	is the program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed time CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value.
Average Not In Use queue membership time		Reset characteristic: reset to zero is the average length of time that a program is eligible for removal from storage by the DPSC mechanism. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value. Reset characteristic: none

Table 91. Loader domain: global statistics (continued)

DFHSTUP name	Field name	Description
Reclaims from Not In Use queue	LDGRNIU	is the number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC). Reset characteristic: reset to zero
Programs loaded but Not In Use	LDGPNIU	is the number of programs on the Not-In-Use (NIU) queue. Reset characteristic: not reset
Amount of DSA occupied by Not In Use programs	LDGCNIU	is the current amount of ERDSA storage which is occupied by Not-In-Use (NIU) programs. Reset characteristic: not reset

Loader domain: summary global statistics

Summary statistics are not available online.

These statistics fields contain the summary global data for the loader.

Table 92. Loader domain: summary global statistics

DFHSTUP name	Description
Library load requests	is the total number of times the loader has issued an MVS LOAD request to load programs from the DFHRPL library concatenation into CICS managed storage. Modules in the LPA are not included in this figure.
Total loading time	is the total time taken for the number of library loads indicated by LDGLLR. The DFHSTUP report expresses this time as *days-hours:minutes:seconds.decimals*.
Average loading time	is the average time to load a program from the DFHRPL library concatenation into CICS managed storage. This value is expressed as *minutes:seconds.decimals*.
Program uses	is the total number of uses of any program by the CICS system.
Requests that waited	is the total number of loader domain requests that were forced to suspend due to the loader domain performing an operation on that program on behalf of another task. These operations could be: • A NEWCOPY request • Searching the LPA • A physical load in progress.
Peak waiting Loader requests	is the peak number of tasks suspended at one time.
Times at peak	is the total number of times the peak level indicated by the previous statistic was reached. This, along with the previous 2 values, is an indication of the level of contention for loader resource.
Total waiting time	is the total suspended time for the number of tasks indicated by the "Requests that waited" statistic. The DFHSTUP report expresses this time as *days-hours:minutes:seconds.decimals*.

Table 92. Loader domain: summary global statistics (continued)

DFHSTUP name	Description
Times DFHRPL re-opened	is the total number of times the loader received an end-of-extent condition during a LOAD and successfully closed and re-opened the DFHRPL library and retried the LOAD.
CDSA	
Programs removed by compression	is the total number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism.
Total Not In Use queue membership time	is the total program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed time CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue. The DFHSTUP report expresses this time as *days-hours:minutes:seconds.decimals*.
Average Not In Use queue membership time	is the average time between a program becoming eligible for removal from storage by the DPSC and the actual time of its removal from storage. This statistic is expressed in *minutes:seconds.decimals*.
Reclaims from Not In Use queue	is the total number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC).
Programs loaded but Not In Use	is the total number of programs on the Not-In-Use (NIU) queue.
ECDSA	
Programs removed by compression	is the total number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism.
Total Not In Use queue membership time	is the total program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed time CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue. The DFHSTUP report expresses this time as *days-hours:minutes:seconds.decimals*.
Average Not In Use queue membership time	is the average time between a program becoming eligible for removal from storage by the DPSC and the actual time of its removal from storage. This statistic is expressed in *minutes:seconds.decimals*.
Reclaims from Not In Use queue	is the total number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC).
Programs loaded but Not In Use	is the total number of programs on the Not-In-Use (NIU) queue.
SDSA	
Programs removed by compression	is the total number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism.

Table 92. Loader domain: summary global statistics (continued)

DFHSTUP name	Description
Total Not In Use queue membership time	is the total program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed time CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue. The DFHSTUP report expresses this time as *days-hours:minutes:seconds.decimals*.
Average Not In Use queue membership time	is the average time between a program becoming eligible for removal from storage by the DPSC and the actual time of its removal from storage. This statistic is expressed in *minutes:seconds.decimals*.
Reclaims from Not In Use queue	is the total number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC).
Programs loaded but Not In Use	is the total number of programs on the Not-In-Use (NIU) queue.
ESDSA	
Programs removed by compression	is the total number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism.
Total Not In Use queue membership time	is the total program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed time CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue. The DFHSTUP report expresses this time as *days-hours:minutes:seconds.decimals*.
Average Not In Use queue membership time	is the average time between a program becoming eligible for removal from storage by the DPSC and the actual time of its removal from storage. This statistic is expressed in *minutes:seconds.decimals*.
Reclaims from Not In Use queue	is the total number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC).
Programs loaded but Not In Use	is the total number of programs on the Not-In-Use (NIU) queue.
ESDSA	
Programs removed by compression	is the total number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism.

Table 92. Loader domain: summary global statistics (continued)

DFHSTUP name	Description
Total Not In Use queue membership time	is the total program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed time CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue. The DFHSTUP report expresses this time as *days-hours:minutes:seconds.decimals*.
Average Not In Use queue membership time	is the average time between a program becoming eligible for removal from storage by the DPSC and the actual time of its removal from storage. This statistic is expressed in *minutes:seconds.decimals*.
Reclaims from Not In Use queue	is the total number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC).
Programs loaded but Not In Use	is the total number of programs on the Not-In-Use (NIU) queue.
RDSA	
Programs removed by compression	is the total number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism.
Total Not In Use queue membership time	is the total program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed time CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue. The DFHSTUP report expresses this time as *days-hours:minutes:seconds.decimals*.
Average Not In Use queue membership time	is the average time between a program becoming eligible for removal from storage by the DPSC and the actual time of its removal from storage. This statistic is expressed in *minutes:seconds.decimals*.
Reclaims from Not In Use queue	is the total number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC).
Programs loaded but Not In Use	is the total number of programs on the Not-In-Use (NIU) queue.
ERDSA	
Programs removed by compression	is the total number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism.

Table 92. Loader domain: summary global statistics (continued)

DFHSTUP name	Description
Total Not In Use queue membership time	is the total program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed time CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue. The DFHSTUP report expresses this time as *days-hours:minutes:seconds.decimals*.
Average Not In Use queue membership time	is the average time between a program becoming eligible for removal from storage by the DPSC and the actual time of its removal from storage. This statistic is expressed in *minutes:seconds.decimals*.
Reclaims from Not In Use queue	is the total number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC).
Programs loaded but Not In Use	is the total number of programs on the Not-In-Use (NIU) queue.

Program

Program: resource statistics

These statistics fields contain the resource data collected by the loader for each program. They are available online, and are mapped by the DFHLDRDS DSECT.

Table 93. Programs: resource statistics

DFHSTUP name	Field name	Description
Program name	LDRPNAME	is the name of the program.
Times used	LDRTU	Reset characteristic: not reset is the number of times CICS tasks within the system have issued load requests to the loader domain to obtain access to a usable instance of this program. These load requests may cause the loader domain to issue an MVS LOAD.
Fetch count	LDRFC	Reset characteristic: reset to zero is the number of times the loader domain has issued an MVS LOAD request to load a copy of the program from the DFHRPL library concatenation into CICS managed storage.
NOT IN THE DFHSTUP REPORT	LDRFT	Reset characteristic: reset to zero is the time taken to perform all fetches. The DSECT field contains a four-byte value that expresses the time in 16-microsecond units. Reset characteristic: reset to zero

Table 93. Programs: resource statistics (continued)

DFHSTUP name	Field name	Description
Average fetch time	Calculated by DFHSTUP	is the average time taken to perform a fetch of the program. The DFHSTUP report expresses this time as *minutes:seconds.decimals*. Reset characteristic: reset to zero
RPL offset	LDRRPLO	is the offset into the DFHRPL DD concatenation of the library from which the program was last loaded or is loaded when next required non-LPA resident modules only. **Note:** The offset values begin with zero for the first partitioned data set in the concatenation and thus this field may not be used to deduce whether a copy of the program is available to the loader domain. Reset characteristic: not reset
NEWCOPY count	LDRTN	is the number of times a NEWCOPY has been requested against this program. Reset characteristic: reset to zero
Program size	LDRPSIZE	is the size of the program in bytes, if known (otherwise zero). Reset characteristic: not reset
Times removed	LDRRPC	is the number of times an instance of this program has been removed from CICS managed storage due to the actions of the Dynamic Program Storage Compression (DPSC) mechanism. Reset characteristic: reset to zero

Table 93. Programs: resource statistics (continued)

DFHSTUP name	Field name	Description
Location	LDRLOCN	is the location of the current storage resident instance of the program, if any. It has one of the following values: **DFHSTUP value** NONE **DSECT value** LDRNOCO (X'00') **Meaning** No current copy **DFHSTUP value** CDSA **DSECT value** LDRCDCO (X'01') **Meaning** Current copy in the CDSA **DFHSTUP value** SDSA **DSECT value** LDRSDCO (X'08') **Meaning** Current copy in the SDSA **DFHSTUP value** LPA **DSECT value** LDRLPACO (X'03') **Meaning** Current copy in the LPA **DFHSTUP value** ECDSA **DSECT value** LDRECDCO (X'04') **Meaning** Current copy in the ECDSA **DFHSTUP value** ESDSA **DSECT value** LDRESDCO (X'09') **Meaning** Current copy in the ESDSA **DFHSTUP value** ERDSA **DSECT value** LDRERDCO (X'06') **Meaning** Current copy in the ERDSA **DFHSTUP value** RDSA **DSECT value** LDRRDCO (X'0A') **Meaning** Current copy in the RDSA Reset characteristic: not reset

Programs: summary resource statistics

Summary statistics are not available online.

These statistics fields contain the summary resource data statistics for the loader for each program.

Table 94. Programs: summary resource statistics

DFHSTUP name	Description
Program name	is the name of the program.
Times used	is the total number of times CICS tasks within the system have issued load requests to the loader domain to obtain access to a usable instance of this program. These load requests may cause the loader domain to issue MVS LOAD requests to obtain access to usable instances of this program.

Table 94. Programs: summary resource statistics (continued)

DFHSTUP name	Description
Fetch count	is the total number of times the loader domain has issued an MVS LOAD request to load a copy of the program from the DFHRPL library concatenation into CICS managed storage.
Average fetch time	is the average time taken to perform a fetch of the program. The DFHSTUP report expresses this time as *minutes:seconds.decimals*.
NEWCOPY count	is the total number of times a NEWCOPY has been requested against this program.
Times removed	is the total number of times an instance of this program has been removed from CICS managed storage due to the actions of the Dynamic Program Storage Compression (DPSC) mechanism.

Recovery manager

Recovery manager: global statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS RECOVERY command, and are mapped by the DFHRMGDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 95. Recovery manager: global statistics

DFHSTUP name	Field name	Description
Total number of syncpoints (forward)	RMGSYFWD	is the total number of syncpoint requests to commit forward. Reset characteristic: reset to zero
Total number of syncpoints (backward)	RMGSYBWD	is the total number of syncpoint requests to commit backward (for example, EXEC CICS SYNCPOINT ROLLBACK). Reset characteristic: reset to zero
Total number of resynchronizations	RMGRESYN	is the total number of resynchronization requests. Reset characteristic: reset to zero
Total UOWs shunted for indoubt failure	RMGTSHIN	is the total number of units of work that lost connection to their recovery coordinator during syncpoint processing and had to be shunted for indoubt failure, but have now completed. Note that this value does not include those units of work that are currently shunted for indoubt failure. Reset characteristic: reset to zero
Total time in shunts for indoubt failure	RMGTSHTI	is the total time (STCK) that the units of work shunted for indoubt failure (RMGTSHIN) spent waiting in this condition, but have now completed. Note that this value does not include those units of work that are currently shunted for indoubt failure. Reset characteristic: reset to zero

Table 95. Recovery manager: global statistics (continued)

DFHSTUP name	Field name	Description
Total UOWs shunted for commit/backout failure	RMGTSHRO	is the total number of units of work that had to be shunted for commit/backout failure because a local resource manager could not perform commit/backout processing at this time on behalf of the UOW during syncpoint, but have now completed. Note that this value does not include those units of work that are currently shunted for commit/backout failure. Reset characteristic: reset to zero
Total time shunted for commit/backout failure	RMGTSHTR	is the total time (STCK) that the units of work shunted for commit/backout (RMGTSHRO) failures spent waiting in this condition, but have now completed. Note that this value does not include those units of work that are currently shunted for commit/backout failure. Reset characteristic: reset to zero
Current UOWs shunted for indoubt failure	RMGCSHIN	is the current number of units of work that lost the connection to their recovery coordinator during syncpoint processing, and have been shunted for indoubt failure. Reset characteristic: reset to zero
Current time in shunts for indoubt failure	RMGCSHTI	is the total time (STCK) that the units of work currently shunted for indoubt failure (RMGCSHIN) have been waiting in this condition so far. Reset characteristic: reset to zero
Current UOWs shunted for resource failure	RMGCHSHR	is the current number of units of work that have been shunted for commit/backout failure because a local resource manager was not able to perform commit/backout processing at this time on behalf of the UOW during syncpoint Reset characteristic: reset to zero
Current time in shunts for resource failure	RMGCSHTR	is the total time (STCK) that the units of work currently shunted for commit/backout (RMGCHSHR) failures have been waiting in this condition so far. Reset characteristic: reset to zero

The following fields detail the reasons why UOWs may have introduced integrity exposures because they were forced to complete prematurely. The UOWs were not allowed to shunt, not capable of shunting, or forced to terminate a shunt, regardless of the outcome.

Table 95. Recovery manager: global statistics (continued)

DFHSTUP name	Field name	Description
Total forces of indoubt action by trandef	RMGIAFTR	is the total number of UOWs that were forced to complete syncpoint processing, despite losing the connection to the recovery coordinator, because their transaction definition specified that they could not wait indoubt. The UOWs would have committed or backed out according to the transaction definition indoubt action attribute, regardless of the actions specified or taken by any other participating region in this distributed UOW. Reset characteristic: reset to zero
Total forces of indoubt action by timeout	RMGIAFTI	is the total number of shunted indoubt UOWs that were forced to complete syncpoint processing, although still unconnected to the recovery coordinator, because their transaction definition wait for indoubt timeout value was exceeded. The UOWs would have committed or backed out according to the transaction definition indoubt action attribute, regardless of the actions specified or taken by any other participating region in this distributed UOW. Reset characteristic: reset to zero
Total forces of indoubt action by operator	RMGIAFOP	is the total number of shunted indoubt UOWs that were forced to complete syncpoint processing, although still unconnected to the recovery coordinator, through a CEMT, or EXEC CICS, SET UOW command forced a resolution. The UOWs would have committed or backed out according to the command option, regardless of the actions specified or taken by any other participating region in this distributed UOW. Reset characteristic: reset to zero

Table 95. Recovery manager: global statistics (continued)

DFHSTUP name	Field name	Description
Total forces of indoubt action by no wait	RMGIAFNW	is the total number of UOWs that were forced to complete syncpoint processing, despite having the ability to wait indoubt, because a local resource owner or connected resource manager used by the UOW was unable to wait indoubt. The UOWs would have committed or backed out according to the transaction definition indoubt action attribute, regardless of the actions specified or taken by any other participating region in this distributed UOW. See the following section on no support for indoubt waiting breakdown. Reset characteristic: reset to zero
Total forces of indoubt action by other	RMGIAFOT	is the total number of UOWs that were forced to complete syncpoint processing, despite having the ability to wait indoubt, because of reasons other than those described above (for example, a cold start of the coordinator, level of RMI adapter modification, and resynchronization errors). The UOWs would have committed or backed out according to the transaction definition indoubt action attribute, regardless of the actions specified or taken by any other participating region in this distributed UOW. Reset characteristic: reset to zero
No support for indoubt waiting breakdown The following fields further detail the reasons why a UOW did not have the ability to wait indoubt (shunt) at the time of indoubt failure (lost coordinator), and are breakdowns of the field RMGIAFNW. This is because the UOW uses either recoverable local resources, recoverable resources across intersystem links, or external resource managers (RMI), which do not have the ability to wait indoubt. As a result of a resolution of a UOW being forced for this reason, integrity exposures may occur.		
–Indoubt action forced by TD queues	RMGNWTD	is the number of UOW forces that occurred because the UOW uses a recoverable transient data queue defined with an indoubt attribute of WAIT=NO. Reset characteristic: reset to zero
–Indoubt action forced by LU61 connections	RMGNW61	is the number of UOW forces that occurred because the UOW uses an LU6.1 intersystem link, which cannot support indoubt waiting. Note that if an LU6.1 intersystem link can operate as last agent in syncpoint processing the lack of waiting ability is immaterial. For more details about last agent processing, see Reset characteristic: reset to zero

Table 95. Recovery manager: global statistics (continued)

DFHSTUP name	Field name	Description
–Indoubt action forced by MRO connections	RMGNWMRO	is the number of UOW forces that occurred because the UOW uses an MRO intersystem link to a downlevel CICS region, which cannot support indoubt waiting. Note that if an MRO intersystem link can operate as last agent in syncpoint processing the lack of waiting ability is immaterial. For more details about last agent processing, see Reset characteristic: reset to zero
–Indoubt action forced by RMI exits (TRUEs)	RMGNWRMI	is the number of UOW forces that occurred because the UOW uses an RMI that declared an interest in syncpoint but could not support indoubt waiting. Note that if an RMI intersystem link can operate as last agent in syncpoint processing the lack of waiting ability is immaterial. For more details about last agent processing, see Reset characteristic: reset to zero
–Indoubt action forced by others	RMGNWOTH	is the number of UOW forces that occurred because the UOW uses recoverable facilities other than above (for example, terminal RDO), which invalidate the ability to support indoubt waiting. Reset characteristic: reset to zero
–Total number of indoubt action mismatches	RMGIAMIS	is the total number of UOWs that were forced to resolve using an indoubt action attribute, whether by definition, option or operator override (as detailed in the above fields), and on so doing detected an indoubt action attribute mismatch with a participating system or RMI. For example, a participating system in a distributed UOW resolves its work forward while other systems back out theirs. The opposite also applies. Reset characteristic: reset to zero

Recovery manager: summary global statistics

Summary statistics are not available online.

Table 96. Recovery manager: summary global statistics

DFHSTUP name	Description
Total number of syncpoints (forward)	is the total number of syncpoint requests to commit forward.
Total number of syncpoints (backward)	is the total number of syncpoint requests to commit backward. For example, EXEC CICS SYNCPOINT ROLLBACK.
Total number of resynchronizations	is the total number of reaynchronization requests.
Total number of shunts for indoubt failure	is the total number of UOWs that have lost connection to their recovery coordinator during syncpoint processing, had to be chunted for indoubt failure, but have now completed.

Table 96. Recovery manager: summary global statistics (continued)

DFHSTUP name	Description
Total time in shunts for indoubt failure	is the total time (STCK) that the UOWs shunted for indoubt failure (RMGTSHIN) spent waiting in this condition.
Total UOWs shunted for commit/backout failure	is the total number of UOWs that had to be shunted for commit/backout failure because a local resource manager was not able to perform commit/backout processing at that time, but have now completed.
Total time shunted for commit/backout failure	is the total time (STCK) that the UOWs shunted for commit/ backout (RMGTSHRO) failures waited in this condition, but have now completed.
Outstanding indoubt failure shunted UOWs	is the current number of UOWs that have been shunted for indoubt failure because the connection to their recovery coordinator during syncpoint processing was lost.
Outstanding shunted UOWs time in shunts	is the total time (STCK) that the UOWs currently shunted for indoubt failure (RMGTSHIN) spent waiting in this condition so far.
Outstanding shunted UOWs for resource failure	is the current number of UOWs that have been shunted for commit/ backout failure because a local resource manager was unable to perform commit/backout processing at that time on behalf of the UOW.
Outstanding shunt time for resource failure	is the total time (STCK) that the UOWs are currently shunted for commit/backout (RMGCHSIR) failures have been waiting in this condition so far.
Total forces of indoubt action by trandef	is the total number of UOWs that were forced to complete syncpoint processing, despite losing the connection to the recovery coordinator, because their transaction definition specifying that they could not wait indoubt.
Total forces of indoubt action by timeout	is the total number of shunted indoubt UOWs that were forced to complete syncpoint processing, although still unconnected to the recovery coordinator, because their transaction definition wait for indoubt timeout value was exceeded.
Total forces of indoubt action by operator	is the total number of shunted indoubt UOWs that were forced to complete syncpoint processing, although still unconnected to the recovery coordinator because the operator (CEMT) forced a resolution.
Total forces of indoubt action by no wait	is the total number of UOWs that were forced to complete syncpoint processing, despite having the ability to wait indoubt because a local resource owner or connected resource manager that the UOW used was unable to wait indoubt.
Total forces of indoubt action by other	is the total number of UOWs
No support for indoubt waiting breakdown	
–Indoubt action forced by TD queues	is the number of UOW forces that occurred because the UOW was using a recoverable transient data queue defined with an indoubt attribute of WAIT=NO.
–Indoubt action forced by LU61 connections	is the number of UOW forces that occurred because the UOW used an LU6.1 intersystem link, which cannot support indoubt waiting.
–Indoubt action forced by MRO connections	is the number of UOW forces that occurred because the UOW using an MRO intersystem link to a downlevel CICS region, which cannot support indoubt waiting.
–Indoubt action forced by RMI exits (TRUEs)	is the number of UOW forces that occurred because the OUW using an RMI that declared an interest in syncpoint but could not support indoubt waiting.
–Indoubt action forced by others	is the number of UOW forces that occurred because the UOW used recoverable facilities other than above, for example, terminal RDO, which invalidates the ability to support indoubt waiting.

Table 96. Recovery manager: summary global statistics (continued)

DFHSTUP name	Description
Total number of indoubt action mismatches	is the total number of UOWs that were forced to resolve using an indoubt action attribute, whether by definition, option, or operator override (as detailed in the above fields), and detected an indoubt action attribute mismatch with a participating system or RMI. For example, a participating system in a distributed UOW resolves its work forward while other systems back out theirs. The opposite also applies.

Statistics domain

Statistics domain: global statistics

These statistics are available online, and are mapped by the DFHSTGDS DSECT.

Table 97. Statistics domain: global statistics

DFHSTUP name	Field name	Description
Interval collections so far	STGNC	is the number of interval collections made during the CICS run, or from one end-of-day to the following end-of-day. Reset characteristic: This field is reset to zero only at every end-of-day collection.
SMF writes	STGSMFW	is the number of SMF writes since the last reset time. This figure includes records written for all types of statistics collections. Reset characteristic: reset to zero
NOT IN THE DFHSTUP REPORT	STGLDW	is the length of data written to SMF during an interval, expressed as bytes. This figure includes length of data written during an interval for unsolicited, requested, and interval/end-of-day collections. Reset characteristic: reset to zero **Note:** This field contains the accumulated length of statistics records excluding the SMF headers.

Interval, end-of-day, and requested statistics all contain the same items.

Statistics domain: summary global statistics

Summary statistics are not available online.

Table 98. Statistics domain: summary global statistics

DFHSTUP name	Description
Total number of interval collections	is the total number of interval collections made during the entire CICS run.

Table 98. Statistics domain: summary global statistics (continued)

DFHSTUP name	Description
Total number of SMF writes	is the total number of SMF writes during the entire CICS run. This figure includes records written during an interval for unsolicited, requested, and interval/end-of-day collections.

Storage manager

These statistics are produced to aid all aspects of storage management.

Storage manager statistics: domain subpools

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS STORAGE command, and are mapped by the DFHSMDDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 99. Storage manager statistics: domain subpools

DFHSTUP name	Field name	Description
Subpool Name	SMDSPN	is the unique 8-character name of the domain subpool. The values of the domain subpool field are described in "Appendix F. MVS and CICS virtual storage" on page 611. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	SMDETYPE	The assembler DSECT field name has the value X'01' or X'02', indicating whether all the elements in the subpool are fixed length or variable length. For further information about subpool elements, see "Appendix F. MVS and CICS virtual storage" on page 611. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	SMDFLEN	is the length of each subpool element (applicable to FIXED length subpools only). For further information about subpool elements, see "Appendix F. MVS and CICS virtual storage" on page 611. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	SMDELCHN	The assembler DSECT field name has the value X'01' or X'02', indicating whether or not SM maintains an element chain for the subpool with the addresses and lengths of each element. For further information about element chains, see "Appendix F. MVS and CICS virtual storage" on page 611. Reset characteristic: not reset

Table 99. Storage manager statistics: domain subpools (continued)

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	SMDBNDRY	is the boundary on which each element is aligned. This is a power of 2 in the range 8 through 4096 bytes. For further information about boundaries, see "Appendix F. MVS and CICS virtual storage" on page 611. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	SMDLOCN	The storage location of this domain subpool. The assembler DSECT field name has the following values: • SMDBELOW (X'01') below the 16MB line. • SMDABOVE (X'02') above the 16MB line. Reset characteristic: not reset
Location	SMDDSANAME	Name of the DSA that the domain subpool is allocated from. Values can be 'CDSA', 'SDSA', 'RDSA', 'ECDSA', 'ESDSA', and 'ERDSA'. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	SMDDSAINDEX	A unique identifier for the dynamic storage area that this subpool is allocated from. Values can be • SMDCDSA (X'01') indicating that the subpool storage is obtained from the CDSA. • SMDSDSA (X'03') indicating that the subpool storage is obtained from the UDSA. • SMDRDSA (X'04') indicating that the subpool storage is obtained from the UDSA. • SMDECDSA (X'05') indicating that the subpool storage is obtained from the ECDSA. • SMDESDSA (X'07') indicating that the subpool storage is obtained from the EUDSA. • SMDERDSA (X'08') indicating that the subpool storage is obtained from the ERDSA. Reset characteristic: not reset
Access	SMDACCESS	is the type of access of the subpool. It will be either CICS, USER, or READONLY. If storage protection not active, all storage areas will revert to CICS except those in the ERDSA. • SMDCICS (X'01') access is CICS key. • SMDUSER (X'02') access is USER key. • SMDREADONLY (X'03') is read-only protection. Reset characteristic: not reset

Table 99. Storage manager statistics: domain subpools (continued)

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	SMDIFREE	is the size of the initial free area for the subpool (which may be zero), expressed in bytes. For further information about the initial free area, see "Appendix F. MVS and CICS virtual storage" on page 611.
Getmain Requests	SMDGMREQ	Reset characteristic: not reset is the number of GETMAIN requests for the subpool.
Freemain Requests	SMDFMREQ	Reset characteristic: reset to zero is the number of FREEMAIN requests for the subpool.
Current Elements	SMDCELEM	Reset characteristic: reset to zero is the current number of storage elements in the subpool.
Current Elem Stg	SMDCES	Reset characteristic: not reset is the sum of the lengths of all the elements in the subpool, expressed in bytes.
Current Page Stg	SMDCPS	Reset characteristic: not reset is the space taken by all the pages allocated to the subpool, expressed in bytes.
Peak Page Stg	SMDHWMPS	Reset characteristic: not reset is the peak page storage allocated to support the storage requirements of this subpool. Reset characteristic: reset to current value

Summary domain subpools statistics

Summary statistics are not available online.

Table 100. Summary domain subpools statistics

DFHSTUP name	Description
Subpool Name	is the unique 8-character name of the domain subpool. The values of the domain subpool field are described in "Appendix F. MVS and CICS virtual storage" on page 611.
Location	is the indicator of the subpool location (CDSA, SDSA, RDSA, ECDSA, ESDSA, or ERDSA).
Access	is the type of access of the subpool. It will be either CICS, USER, or READONLY. If storage protection not active, all storage areas will revert to CICS except those in the ERDSA.
Getmain Requests	is the total number of GETMAIN requests for the subpool.
Freemain Requests	is the total number of FREEMAIN requests for the subpool.
Peak Elements	is the peak number of storage elements in the subpool.
Peak Elem Stg	is the peak amount of element storage in the subpool, expressed in bytes.
Peak Page Stg	is the peak amount of page storage in the subpool, expressed in bytes.

Storage manager: global statistics

These statistics ***can*** be accessed online using the EXEC CICS COLLECT STATISTICS command. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

These statistics are collected for each pagepool. They are available online, and are mapped by the DFHSMSDS DSECT.

Table 101. Storage manager: global statistics

DFHSTUP name	Field name	Description
Storage protection	SMSSTGPROT	X'01' active X'00' not active
Transaction isolation	SMSTRANISO	X'01' active X'00' not active
Reentrant programs	SMSRENTPGM	X'01' protect. RDSA and ERDSA obtained from key 0 storage. X'00' no protect. RDSA and ERDSA obtained from key 8 storage.
Current DSA limit	SMSDSALIMIT	Current DSA limit value
Current DSA total	SMSDSATOTAL	Total amount of storage currently allocated to the DSAs below the line. This value may be smaller or larger than SMSDSALIMIT.
Peak DSA total	SMSHWMDSATOTAL	The total amount of storage allocated to the DSAs below the line. This value may be smaller or larger than SMSDSALIMIT.
Current EDSDA limit	SMSEDSALIMIT	Current EDSA limit
Current EDSA total	SMSEDSATOTAL	Total amount of storage currently allocated to the DSAs above the line. This value may be smaller or larger than EDSALIMIT.
Peak EDSA total	SMSHWMESDATOTAL	The total amount of storage allocated to the DSAs above the line. This value may be smaller or larger than SMSESDALIMIT.

Storage manager: subspace statistics

These statistics ***can*** be accessed online using the EXEC CICS COLLECT STATISTICS command. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

These statistics are collected for each pagepool, and are mapped by the DFHSMSDS DSECT.

Table 102. Storage manager: subspace statistics

DFHSTUP name	Field name	Description
Current unique subspace users	SMSUSSCUR	Current number of unique subspace users. Number of tasks currently allocated a unique subspace.
Total unique subspace users	SMSUSSCUM	Total number of tasks that have been allocated a unique subspace.
Peak unique subspace users	SMSUSSHWM	The peak number of tasks concurrently allocated a unique subspace.
Current common subspace users	SMSCSSCUR	Number of tasks currently allocated to the common subspace
Total common subspace users	SMSCSSCUM	Total number of tasks allocated to the common subspace
Peak common subspace users	SMSCSSHWM	The peak number of tasks concurrently allocated to the common subspace.

Storage manager statistics: dynamic storage areas

These statistics ***can*** be accessed online using the EXEC CICS COLLECT STATISTICS command. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

These statistics are collected for each pagepool. They are available online, and are mapped by the DFHSMSDS DSECT.

Table 103. Storage manager statistics: dynamic storage areas

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	SMSNPAGP	is the number of pagepools in the CICS region. There are eight pagepools: the CDSA (CICS dynamic storage area), the UDSA (user dynamic storage area), the SDSA (shared dynamic storage area), the RDSA (read-only dynamic storage area), the ECDSA (extended CICS dynamic storage area), the ESDSA (extended shared dynamic storage area), the EUDSA (extended user dynamic storage area), and the ERDSA (extended read-only dynamic storage area). Reset characteristic: not reset

Note: The following fields are mapped by the SMSBODY DSECT within the DFHSMSDS DSECT. The SMSBODY DSECT is repeated for each pagepool in the CICS region (SMSNPAGP).

Table 103. Storage manager statistics: dynamic storage areas (continued)

DFHSTUP name	Field name	Description
Header in DFHSTUP report	SMSDSANAME	Name of the DSA that this record represents. Values can be 'CDSA', 'UDSA', 'SDSA', 'RDSA', 'ECDSA', 'EUDSA', 'ESDSA', and 'ERDSA'.
NOT IN THE DFHSTUP REPORT	SMSDSAINDEX	Reset characteristic: not reset A unique identifier for the dynamic storage area that this subpool is allocated from. Values can be: • SMSCDSA (X'01') The page pool is the CDSA. • SMSUDSA (X'02') The page pool is the UDSA. • SMSSDSA (X'03') The page pool is the SDSA. • SMSRDSA (X'04') The page pool is the RDSA. • SMSECDSA (X'05') The page pool is the ECDSA. • SMSEUDSA (X'06') The page pool is the EUDSA. • SMSESDSA (X'07') The page pool is the ESDSA. • SMSERDSA (X'08') The page pool is the ERDSA.
NOT IN THE DFHSTUP REPORT	SMSLOCN	Reset characteristic: not reset is the location of this pagepool. The assembler DSECT field name has the following values: • SMSBELOW (X'01') below the 16MB line • SMSABOVE (X'02') above the 16MB line.
Current DSA Size	SMSDSASZ	is the current size of the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, or ERDSA, expressed in bytes.
Peak DSA Size	SMSHWMDSASZ	Reset characteristic: not reset is the peak size of the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, or ERDSA, expressed in bytes sinces that last time that statistics were recorded.
Cushion Size	SMSCSIZE	Reset characteristic: not reset is the size of the cushion, expressed in bytes. The cushion forms part of the CDSA, UDSA, ECDSA, EUDSA, or the ERDSA, and is the amount of storage below which CICS goes SOS. Reset characteristic: not reset

Table 103. Storage manager statistics: dynamic storage areas (continued)

DFHSTUP name	Field name	Description
Free storage (inc. cushion)	SMSFSTG	is the amount of free storage in this pagepool, that is the number of free pages multiplied by the page size (4K), expressed in bytes.
Percentage free storage		Reset characteristic: not reset is the percentage of the storage that is free. This value is calculated offline by DFHSTUP and is, therefore, not accessible via the EXEC CICS COLLECT STATISTICS command.
Peak free storage	SMSHWMFSTG	Reset characteristic: not reset is the largest amount of storage that is free since the last time that statistics were recorded.
Lowest free storage	SMSLWMFSTG	Reset characteristic: not reset is the smallest amount of storage that is free since the last time that statistics were recorded.
Largest free area	SMSLFA	Reset characteristic: not reset is the length of the largest contiguous free area in the CDSA RDSA, SDSA, EDSA, UDSA, ECDSA, EUDSA, or ERDSA, expressed in bytes. To get an indication of the storage fragmentation in this pagepool, compare this value with "Free storage" (SMSFSTG) in the pagepool. If the ratio is large, this pagepool is fragmented.
Getmain Requests	SMSGMREQ	Reset characteristic: not reset is the number of GETMAIN requests from the CDSA, RDSA, SDSA, EDSA, UDSA, or ECDSA, EUDSA, or ERDSA.
Freemain Requests	SMSFMREQ	Reset characteristic: reset to zero is the number of FREEMAIN requests from the CDSA, UDSA, ECDSA, EUDSA, RDSA, SDSA, EDSA, or ERDSA.
NOT IN THE DFHSTUP REPORT	SMSASR	Reset characteristic: reset to zero is the number of ADD_SUBPOOL requests to create a subpool (domain or task) from the CDSA, UDSA, ECDSA, EUDSA, ERDSA, RDSA, SDSA, or ESDSA.
		Reset characteristic: reset to zero

Table 103. Storage manager statistics: dynamic storage areas (continued)

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	SMSDSR	is the number of DELETE_SUBPOOL requests (domain or task) from the CDSA, UDSA, ECDSA, EUDSA, ERDSA, RDSA, SDSA, or ESDSA. Reset characteristic: reset to zero
NOT IN THE DFHSTUP REPORT	SMSCSUBP	is the current number of subpools (domain and task) in the CDSA, UDSA, ECDSA, EUDSA, ERDSA, RDSA, SDSA, or ESDSA. Reset characteristic: not reset
Times no storage returned	SMSCRISS	is the number of times a GETMAIN request with SUSPEND(NO) returned the condition INSUFFICIENT_STORAGE. Reset characteristic: reset to zero
Times request suspended	SMSUCSS	is the number of times a GETMAIN request with SUSPEND(YES) was suspended because of insufficient storage to satisfy the request at the moment. Reset characteristic: reset to zero
Current suspended	SMSCSS	is the number of GETMAIN requests currently suspended for storage. Reset characteristic: not reset
Peak requests suspended	SMSHWMSS	is the peak number of GETMAIN requests suspended for storage. Reset characteristic: reset to current value
Purged while waiting	SMSPWWS	is the number of requests which were purged while suspended for storage. Reset characteristic: reset to zero
Times cushion released	SMSCREL	is the number of times a GETMAIN request caused the storage cushion to be released. The cushion is said to be released when the number of free pages drops below the number of pages in the cushion Reset characteristic: reset to zero
Times went short on storage	SMSSOS	is the number of times CICS went SOS in this pagepool (CDSA, UDSA, ECDSA, EUDSA, RDSA, SDSA, ESDSA or ERDSA), where SOS means either that the cushion is currently in use and/or there is at least one task suspended for storage. Reset characteristic: reset to zero

Table 103. Storage manager statistics: dynamic storage areas (continued)

DFHSTUP name	Field name	Description
Total time SOS	SMSTSOS	is the accumulated time that CICS has been SOS in this DSA. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value.
Storage violations	SMSSV	Reset characteristic: reset to zero is the number of storage violations recorded in the CDSA, UDSA, ECDSA, EUDSA, RDSA, SDSA, ESDSA, and the ERDSA.
Access	SMSACCESS	is the type of access of the page subpool. It will be either CICS, USER, or READONLY. If storage protection not active, all storage areas will revert to CICS except those in the ERDSA. • SMSCICS (X'01') access is CICS key. • SMSUSER (X'02') access is USER key. • SMSREADONLY (X'03') is read-only protection.
Current extents	SMSEXTS	Reset characteristic: not reset is the number of extents currently allocated to a specified dynamic storage area.
Extents added	SMSEXTSA	Reset characteristic: not reset is the number of extents added to a dynamic storage area since the last time statistics were recorded.
Extents released	SMSEXTSR	Reset characteristic: not reset is the number of extents which have been released from a dynamic storage area since the last time statistics were recorded.
		Reset characteristic: not reset

Storage manager: summary global statistics

Summary statistics are not available online.

Table 104. Storage manager: summary global statistics

DFHSTUP name	Description
Storage protection	is the total storage protection defined to the storage manager.
Transaction isolation	is the number of transactions associated with this terminal that can be isolated.
Reentrant programs	is the number of programs that have used the reentry facility.
Current DSA limit	this is the limit of CICS dynamic storage area that can be defined by the storage manager.

Table 104. Storage manager: summary global statistics (continued)

DFHSTUP name	Description
Current DSA total	this is the number of CICS dynamic storage areas currently in use by the storage manager.
Peak DSA total	is the highest number of CICS dynamic storage areas used by the storage manager since the last recorded statistics.
Current EDSA limit	this is the number of extended dynamic storage areas currently defined by the storage manager.
Current EDSA total	this is the number of extended dynamic storage areas currently in use by the storage manager.
Peak EDSA total	is the highest number of extended dynamic storage area defined by the storage manager since the last recorded statistics.

Storage manager: summary subspace statistics

Summary statistics are not available online.

Table 105. Storage manager: summary subspace statistics

DFHSTUP name	Description	
Total unique subspace users	SMSUSSCUM	Total number of tasks that have been allocated a unique subspace.
Peak unique subspace users	SMSUSSHWM	The peak number of tasks concurrently allocated a unique subspace.
Total common subspace users	SMSCSSCUM	Total number of tasks allocated to the common subspace.
Peak common subspace users	SMSCSSHWM	The peak number of tasks concurrently allocated to the common subspace.

Summary dynamic storage areas statistics

Summary statistics are not available online.

Table 106. Summary dynamic storage areas statistics

DFHSTUP name	Description
DSA size	is the total size of the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, or ERDSA, expressed in bytes.
Cushion size	is the size of the cushion, expressed in bytes. The cushion forms part of the DSA or the EDSA, and is the amount of storage below which CICS goes SOS.
Getmain requests	is the total number of GETMAIN requests from the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, or ERDSA.
Freemain requests	is the total number of FREEMAIN requests from the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, or ERDSA.
Times no storage returned	is the total number of times a GETMAIN request with SUSPEND(NO) returned the condition INSUFFICIENT_STORAGE.
Times request suspended	is the total number of times a GETMAIN request with SUSPEND(YES) was suspended because of insufficient storage to satisfy the request at the moment.
Peak requests suspended	is the peak number of GETMAIN requests suspended for storage.
Purged while waiting	is the total number of requests which were purged while suspended for storage.
Times cushion released	is the total number of times a GETMAIN request caused the storage cushion to be released. The cushion is said to be released when the number of free pages drops below the number of pages in the cushion

Table 106. Summary dynamic storage areas statistics (continued)

DFHSTUP name	Description
Times went short on storage	is the total number of times CICS went SOS in this pagepool (CDSA, UDSA, ECDSA, EUDSA, or ERDSA), where SOS means either that the cushion is currently in use and/or there is at least one task suspended for storage.
Total time SOS	is the accumulated time that CICS has been SOS in this DSA. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value.
Storage violations	is the total number of storage violations recorded in the CDSA, UDSA, ECDSA, EUDSA, and the ERDSA.
Access	is the type of access of the page subpool. It will be either CICS, USER, or READONLY. If storage protection not active, all storage areas will revert to CICS except those in the ERDSA.

Storage manager statistics: Task subpools

These statistics cannot be accessed online using the EXEC CICS COLLECT STATISTICS command. They are produced only for offline processing (written to SMF).

These statistics are collected for each pagepool. They are mapped by the DFHSMTDS DSECT.

Although task subpools are dynamically created and deleted for each task in the system, these statistics are the sum of all task subpool figures for the task related pagepools (CDSA, UDSA, ECDSA, and EUDSA). If further granularity of task storage usage is required, use the performance class data of the CICS monitoring facility.

Table 107. Storage manager statistics: Task subpools

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	SMTNTASK	is the number of task subpools in the CICS region. Reset characteristic: not reset

Note: The following fields are mapped by the SMTBODY DSECT within the DFHSMTDS DSECT. The SMTBODY DSECT is repeated for each task subpool in the CICS region (SMTNTASK).

DSA Name	SMTDSANAME	Name of the dynamic storage area from which this task storage has been allocated. Values can be 'CDSA', 'UDSA', 'ECDSA',and 'EUDSA'. Reset characteristic: not reset

Table 107. Storage manager statistics: Task subpools (continued)

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	SMTDSAINDEX	A unique identifier for the dynamic storage area that these statistics refer to. Values can be: • SMTCDSA (X'01') indicating that the task storage is obtained from the CDSA • SMTUDSA (X'02') indicating that the task storage is obtained from the UDSA • SMTECDSA (X'05') indicating that the task storage is obtained from the ECDSA • SMTEUDSA (X'06') indicating that the task storage is obtained from the EUDSA Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	SMTLOCN	tells you whether the dynamic storage area is above or below the line. • SMTBELOW (X'01') below the 16MB line • SMTABOVE (X'02') above the 16MB line. Reset characteristic: not reset
Access	SMTACCESS	is the type of access of the subpool. It will be either CICS or USER. • SMTCICS (X'01') access is CICS key • SMTUSER (X'02') access is USER key. Reset characteristic: not reset
Getmain Requests	SMTGMREQ	is the number of task subpool GETMAIN requests from this dynamic storage area. Reset characteristic: not reset
Freemain Requests	SMTFMREQ	is the number of task subpool FREEMAIN requests from this dynamic storage area. Reset characteristic: reset to zero
Current Elements	SMTCNE	is the number of elements in all the task subpools in this dynamic storage area. Reset characteristic: reset to zero
Current Elem Stg	SMTCES	is the sum of the storage occupied by all elements in task subpools within this dynamic storage area, expressed in bytes. Reset characteristic: not reset
Current Page Stg	SMTCPS	is the sum of the storage in all pages allocated to task subpools within this dynamic storage area, expressed in bytes. Reset characteristic: not reset
Peak Page Stg	SMTHWMPS	is the peak page storage allocated to support task storage activity in that dynamic storage area. Reset characteristic: reset to current value

Summary task subpools statistics

Summary statistics are not available online.

The following fields are mapped by the SMTBODY DSECT within the DFHSMTDS DSECT. The SMTBODY DSECT is repeated for each task subpool in the CICS region (SMTNTASK).

Table 108. Summary task subpools statistics

DFHSTUP name	Description
DSA Name	tells you whether the dynamic storage area is in the CDSA, UDSA, ECDSA, or EUDSA.
Access	is the type of access of the subpool. It will be either CICS, or USER.
Getmain Requests	is the total number of task subpool GETMAIN requests from this dynamic storage area.
Freemain Requests	is the total number of task subpool FREEMAIN requests from this dynamic storage area.
Peak Elements	is the peak number of elements in all the task subpools in this dynamic storage area.
Peak Elem Storage	is the peak amount of storage occupied by all elements in task subpools within this dynamic storage area, expressed in bytes.
Peak Page Storage	is the peak amount of storage in all pages allocated to task subpools within this dynamic storage area, expressed in bytes.

Table manager

Table manager: global statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS TABLEMGR command, and are mapped by the DFHA16DS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 109. Table manager: global statistics

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	A16NTAB	is the number of tables defined to the table manager. Reset characteristic: not reset
The following fields are mapped by the A16STATS DSECT, which is repeated for each table (A16NTAB).		
Table Name	A16TNAM	is the name of a CICS table supported by the table manager. Reset characteristic: not reset
Total Size of Table Manager Storage (bytes)	A16TSIZE	is the amount of storage, expressed in bytes, used by the table manager to support the table named in the field above (for example, for scatter tables and directory segments). This does not include storage used by the tables themselves. Reset characteristic: not reset

Table manager: Summary global statistics

Summary statistics are not available online.

Table 110. Table manager: Summary global statistics

DFHSTUP name	Description
Table Name	is the name of a CICS table supported by the table manager.

Table 110. Table manager: Summary global statistics (continued)

DFHSTUP name	Description
Average Table Size (bytes)	is the average amount of storage, expressed in bytes, used by the table manager to support the table named in the field above (for example, for scatter tables and directory segments). This does not include storage used by the tables themselves.
Peak Table Size (bytes)	is the peak amount of storage, expressed in bytes, used by the table manager to support the table named in the field above (for example, for scatter tables and directory segments). This does not include storage used by the tables themselves.

TCP/IP Services - resource statistics

TCP/IP Services: statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS TCPIPSERVICE command, and are mapped by the TCPIPSERVICE and the DFHSORDS dsect. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 111. TCP/IP Services: resource statistics

DFHSTUP name	Field name	Description
TCP/IP Service	SOR_SERVICE_NAME	is the name of the TCP/IP service
Port Number	SOR_PORT_NUMBER	Reset characteristic: not reset is the port number being used for this TCP/IP service.
IP Address	SOR_IP_ADDRESS	Reset characteristic: not reset is the IP address defined for the TCP/IP stack used for this TCP/IP service.
SSL	SOR_SSL_SUPPORT	Reset characteristic: not reset is the level of SSL support defined for this TCP/IP service.
Port Backlog	SOR_BACKLOG	Reset characteristic: not reset is the port backlog for this TCP/IP service.
		Reset characteristic: not reset

Table 111. TCP/IP Services: resource statistics (continued)

DFHSTUP name	Field name	Description
Date Opened	SOR_OPEN_LOCAL	is the date on which this TCP/IP service was opened. If this field is not set, SOR_OPEN_LOCAL contains the hexadecimal value X'0000000000000000', shown in the report as "CLOSED". If the field is set, it contains a date expressed in *mm/dd/yyyy* format. This field contains a valid date if: • The TCP/IP service was open at the time the statistics were taken. • This is an unsolicited statistics request due to the TCP/IP service being closed. Reset characteristic: not reset
Time Opened	SOR_OPEN_LOCAL	is the time at which this TCP/IP service was opened. If this field is not set, SOR_OPEN_LOCAL contains the hexadecimal value X'0000000000000000', shown in the report as "CLOSED". If the field is set, it contains a time expressed as a store clock (STCK) value in local time. This field contains a valid time if: • The TCP/IP service was open at the time the statistics were taken. • This is an unsolicited statistics request due to the TCP/IP service being closed. Reset characteristic: not reset
Date Closed	SOR_CLOSE_LOCAL	is the date on which this TCP/IP service was closed. If this field is not set, SOR_CLOSE_LOCAL contains the hexadecimal value X'0000000000000000', shown in the report as "OPEN". If the field is set, it contains a date expressed in *mm/dd/yyyy* format. Reset characteristic: not reset
Time Closed	SOR_CLOSE_LOCAL	is the time at which this TCP/IP service was closed. If this field is not set, SOR_CLOSE_LOCAL contains the hexadecimal value X'0000000000000000', shown in the report as "OPEN". If the field is set, it contains a time expressed as a store clock (STCK) value in local time. Reset characteristic: not reset

TCP/IP: Summary resource statistics

Summary statistics are not available online.

Table 112. TCP/IP: Summary resource statistics

DFHSTUP name	Description
TCP/IP Service	is the name of the TCP/IP service
Port Number	is the port number being used for this TCP/IP service
IP Address	is the IP address defined for the TCP/IP stack used for this TCP/IP service.
SSL	is the level of SSL support defined for this TCP/IP service.
Port Backlog	is the port backlog defined for this TCP/IP service.

TCP/IP Services - request statistics

TCP/IP Services: request statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS TCPIPSERVICE command are mapped by the TCPIPSERVICE and the DFHSORDS dsect. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 113. Table manager: global statistics

DFHSTUP name	Field name	Description
TCP/IP Service	SOR_SERVICE_NAME	is the name of the TCP/IP service
Port Number	SOR_PORT_NUMBER	Reset characteristic: not reset is the port number being used for this TCP/IP service.
IP Address	SOR_IP_ADDRESS	Reset characteristic: not reset is the IP address of this TCP/IP service.
Current	SOR_CURRENT_CONNS	Reset characteristic: not reset is the current number of connections for the TCP/IP service.
Peak	SOR_PEAK_CONNS	Reset characteristic: reset to zero is the peak of those current.
Trans Attached	SOR_TRANS_ATTACHED	Reset characteristic: reset to zero is the number of transactions attached by this TCP/IP Service.
Receive Requests	SOR_RECEIVES	Reset characteristic: not reset is the number of receive requests issued for the TCP/IP Service.
Bytes Received	SOR_BYTES_RECEIVED	Reset characteristic: reset to zero is the number of bytes received for the TCP/IP service.
		Reset characteristic: reset to zero

Table 113. Table manager: global statistics (continued)

DFHSTUP name	Field name	Description
Average/ Receive	SOR_BYTES_RECEIVED/SOR_RECEIVES	is the average number of bytes per receive request for the TCP/IP service. Reset characteristic: not reset
Send Requests	SOR_SENDS	is the number of send requests issued for the TCP/IP Service. Reset characteristic: reset to zero
Bytes Sent	SOR_BYTES_SENT	is the number of bytes sent for the TCP/IP service. Reset characteristic: reset to zero
Average/Send	SOR_BYTES_SENT/SOR_SENDS	is the average number of bytes per send request for the TCP/IP service. Reset characteristic: not reset

TCP/IP: Summary request statistics

Summary statistics are not available online.

Table 114. TCP/IP: Summary request statistics

DFHSTUP name	Description
TCP/IP Service	is the name of the TCP/IP service.
Port Number	is the port number for the TCP/IP Service.
IP Address	is the IP address for the TCP/IP Service.
Peak Connections	is the peak number of connections for the TCP/IP Service.
Trans Attached	is the total number of transactions attached for the TCP/IP Service.
Receive Requests	is the total number of receive requests issued for the TCP/IP Service.
Bytes Received	is the total number of bytes received for the TCP/IP Service.
Send Requests	is the total number of send requests issued for the TCP/IP Service.
Bytes Sent	is the total number of bytes sent for the TCP/IP Service.

Temporary storage

Temporary storage statistics are produced for the data that is written into a temporary storage queue.

This is the DFHSTUP listing for temporary storage statistics.

Temporary storage: global statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS TSQUEUE command, and are mapped by the DFHTSGDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 115. Temporary storage: global statistics

DFHSTUP name	Field name	Description
Put/Putq main storage requests	TSGSTA5F	is the number of records that application programs wrote to main temporary storage. Reset characteristic: reset to zero
Get/Getq main storage requests	TSGNMG	is the number of records that application programs obtained from main temporary storage. Reset characteristic: reset to zero
Peak storage for temp. storage (main)	TSGSTA6F	is the peak value, expressed in bytes, of the amount of virtual storage used for temporary storage records. Reset characteristic: reset to current value
Current storage for temp. storage (main)	TSGSTA6A	is the current value, expressed in bytes, of the amount of virtual storage used for temporary storage records. Reset characteristic: not reset
Put/Putq auxiliary storage requests	TSGSTA7F	is the number of records that application programs wrote to auxiliary temporary storage. Reset characteristic: reset to zero
Get/Getq auxiliary storage requests	TSGNAG	is the number of records that application programs obtained from auxiliary temporary storage. Reset characteristic: reset to zero
Peak temporary storage names in use	TSGQNUMH	is the peak number of temporary storage queue names in use at any one time. Reset characteristic: reset to current value
Current temporary storage names in use	TSGQNUM	is the current number of temporary storage queue names in use. Reset characteristic: not reset
Number of entries in longest queue	TSGQINH	is the peak number of items in any one queue. Reset characteristic: reset to zero
Times queues created	TSGSTA3F	is the number of times that CICS created individual temporary storage queues. Reset characteristic: reset to zero
Control interval size	TSGCSZ	is the size of VSAM's unit of transmission between DASD and main storage, specified in the CONTROLINTERVALSIZE parameter in the VSAM CLUSTER definition for the temporary storage data set (for guidance information about this, see the *CICS Operations and Utilities Guide*). In general, using large CIs permits more data to be transferred at one time, resulting in less system overhead. Reset characteristic: not reset

Table 115. Temporary storage: global statistics (continued)

DFHSTUP name	Field name	Description
Available bytes per control interval	TSGNAVB	is the number of bytes available for use in the TS data set control interval. Reset characteristic: not reset
Segments per control interval	TSGSPCI	is the number of segments available in the TS control interval. Reset characteristic: not reset
Bytes per segment	TSGBPSEG	is the number of bytes per segment of the TS data set. Reset characteristic: not reset
Writes more than control interval	TSGSTABF	is the number of writes of records whose length was greater than the control interval (CI) size. Reset characteristic: not reset
Longest auxiliary temp storage record	TSGLAR	is the size, expressed in bytes, of the longest record written to the temporary storage data set. Reset characteristic: reset to zero
Number of control intervals available	TSGNCI	is the number of control intervals (CIs) available for auxiliary storage. This is the total available space on the temporary storage data set expressed as a number of control intervals. This is not the space remaining at termination. Reset characteristic: not reset
Peak control intervals in use	TSGNCIAH	is the peak number of CIs containing active data. Reset characteristic: not reset
Current control intervals in use	TSGNCIA	is the current number of CIs containing active data. Reset characteristic: reset to current value
Times aux. storage exhausted	TSGSTA8F	is the number of situations where one or more transactions may have been suspended because of a NOSPACE condition, or (using a HANDLE CONDITION NOSPACE command) may have been forced to abend. Reset characteristic: not reset
Number of temp. storage compressions	TSGSTA9F	is the number of times that the temporary storage buffers were compressed. Reset characteristic: reset to zero
Note: The following statistics are produced for buffer usage:		Reset characteristic: reset to zero
Temporary storage buffers	TSGNBCA	is the number of temporary storage buffers specified in the TS= system initialization parameter or in the overrides. The number of buffers allocated may exceed the number requested. Reset characteristic: not reset

Table 115. Temporary storage: global statistics (continued)

DFHSTUP name	Field name	Description
Buffer waits	TSGBWTN	is the number of times a request was queued because all buffers were allocated to other tasks. A buffer wait occurs if the required control interval is already in a locked buffer, and therefore unavailable, even if there are other buffers available. Reset characteristic: reset to zero
Peak users waiting on buffer	TSGBUWTH	is the peak number of requests queued because no buffers were available. Reset characteristic: reset to current value
Current users waiting on buffer	TSGBUWT	is the current number of requests queued because no buffers were available. Reset characteristic: not reset
Buffer writes	TSGTWTN	is the number of WRITEs to the temporary storage data set. This includes both WRITEs necessitated by recovery requirements (see next item) and WRITEs forced by the buffer being needed to accommodate another CI. Reset characteristic: reset to zero
Forced writes for recovery	TSGTWTNR	is the subset of the total number of WRITEs caused by recovery being specified for queues. This I/O activity is not affected by buffer allocation. Reset characteristic: reset to zero
Buffer reads	TSGTRDN	is the number of times a CI has to be read from disk. Increasing the buffer allocation decreases this activity. Reset characteristic: reset to zero
Format writes	TSGTWTNF	is the number of times a new CI was successfully written at the end of the data set to increase the amount of available space in the data set. A formatted write is attempted only if the current number of CIs available in the auxiliary data set have all been used. Reset characteristic: reset to zero
Note: The following statistics are produced for string usage:		
Temporary storage strings	TSGNVCA	is the number of temporary storage strings specified in the TS= system initialization parameter or in the overrides. The number of strings allocated may exceed the number requested. Reset characteristic: not reset
Peak number of strings in use	TSGNVCAH	is the peak number of concurrent I/O operations. If this is significantly less than the number specified in the SIT, consider reducing the SIT value to approach this number. Reset characteristic: reset to current value

Table 115. Temporary storage: global statistics (continued)

DFHSTUP name	Field name	Description
Times string wait occurred	TSGVWTN	is the number of I/O requests that were queued because no strings were available. This is zero if the number of strings is the same as the number of buffers. If this is a high percentage (over 30%) of the number of I/O requests, consider increasing the number of strings initially allocated. Reset characteristic: reset to zero
Peak number of users waiting on string	TSGVUWTH	is the peak number of I/O requests that were queued at any one time because all strings were in use. Reset characteristic: reset to current value
Current users waiting on string	TSGVUWT	is the current number of I/O requests that are queued because all strings are in use. Reset characteristic: not reset
I/O errors on TS data set	TSGSTAAF	is the number of input/output errors which occurred on the temporary storage data set. This should normally be zero. If it is not, inspect the CICS and VSAM messages to determine the cause. Reset characteristic: reset to zero
Shared pools defined	TSGSHPDF	is the number of shared TS pools defined in the TST. Reset characteristic: reset to zero
Shared pools currently connected	TSGSHPCN	is the number of shared TS pools currently connected to. Reset characteristic: reset to zero
Shared read requests	TSGSHRDS	is the number of read requests to all shared TS pools. Reset characteristic: reset to zero
Shared write requests	TSGSHWTS	is the number of write requests to all shared TS pools. Reset characteristic: reset to zero

Temporary storage: summary global statistics

Summary statistics are not available online.

Table 116. Temporary storage: summary global statistics

DFHSTUP name	Description
Put/Putq main storage requests	is the total number of records that application programs wrote to main temporary storage.
Get/Getq main storage requests	is the total number of records that application programs obtained from main temporary storage.
Peak storage for temp. storage (main)	is the peak value, expressed in bytes, of the amount of virtual storage used for temporary storage records.
Put/Putq auxiliary storage requests	is the total number of records that application programs wrote to auxiliary temporary storage.
Get/Getq auxiliary storage requests	is the total number of records that application programs obtained from auxiliary temporary storage.

Table 116. Temporary storage: summary global statistics (continued)

DFHSTUP name	Description
Peak temporary storage names in use	is the peak number of temporary storage queue names at any one time.
Number of entries in longest queue	is the peak number of items in any one queue, up to a maximum of 32767
Times queues created	is the total number of times that CICS created individual temporary storage queues.
Control interval size	is the size of VSAM's unit of transmission between DASD and main storage, specified in the CONTROLINTERVALSIZE parameter in the VSAM CLUSTER definition for the temporary storage data set (for guidance information about this, see the *CICS Operations and Utilities Guide*). In general, using large CIs permits more data to be transferred at one time, resulting in less system overhead.
Available bytes per control interval	is the number of bytes available for use in each TS data set control interval.
Segments per control interval	is the number of segments in each TS data set control interval.
Bytes per segment	is the number of bytes per segment.
Writes more than control interval	is the total number of writes of records whose length was greater than the control interval (CI) size. If the reported value is large, increase the CI size. If the value is zero, consider reducing the CI size until a small value is reported.
Longest auxiliary temporary storage record	is the size, expressed in bytes, of the longest record written to the temporary storage data set.
Number of control intervals available	is the number of control intervals (CIs) available for auxiliary storage. This is the total available space on the temporary storage data set expressed as a number of control intervals. This is not the space remaining at termination.
Peak control intervals available	is the peak number of CIs containing active data.
Times aux. storage exhausted	is the total number of situations where one or more transactions may have been suspended because of a NOSPACE condition, or (using a HANDLE CONDITION NOSPACE command) may have been forced to abend. If this item appears in the statistics, increase the size of the temporary storage data set.
Number of temp. storage compressions	is the total number of times that temporary storage buffers were compressed.
Note: The following statistics are produced for buffer usage:	
Temporary storage buffers	is the total number of temporary storage buffers specified in the TS= system initialization parameter or in the overrides.
Buffer waits	is the total number of times a request was queued because all buffers were allocated to other tasks. A buffer wait occurs if the required control interval is already in a locked buffer, and therefore unavailable, even if there are other buffers available.
Peak users waiting on buffers	is the peak number of requests queued because no buffers were available.
Buffer writes	is the total number of WRITEs to the temporary storage data set. This includes both WRITEs necessitated by recovery requirements (see next item) and WRITEs forced by the buffer being needed to accommodate another CI. I/O activity caused by the latter reason can be minimized by increasing buffer allocation.
Forced writes for recovery	is the subset of the total number of WRITEs caused by recovery being specified for queues. This I/O activity is not affected by buffer allocation.

Table 116. Temporary storage: summary global statistics (continued)

DFHSTUP name	Description
Buffer reads	is the total number of times a CI has to be read from disk. Increasing the buffer allocation decreases this activity.
Format writes	is the total number of times a new CI was successfully written at the end of the data set to increase the amount of available space in the data set. A formatted write is attempted only if the current number of CIs available in the auxiliary data set have all been used.
Note: The following statistics are produced for string usage:	
Temporary storage strings	is the total number of temporary storage strings specified in the TS= system initialization parameter or in the overrides.
Peak number of strings in use	is the peak number of concurrent I/O operations. If this is significantly less than the number specified in the SIT, consider reducing the SIT value to approach this number.
Times string wait occurred	is the total number of I/O requests that were queued because no strings were available. This is zero if the number of strings is the same as the number of buffers. If this is a high percentage (over 30%) of the number of I/O requests, consider increasing the number of strings initially allocated.
Peak number of users waiting on string	is the peak number of I/O requests that were queued at any one time because all strings were in use.
I/O errors on TS data set	is the total number of input/output errors which occurred on the temporary storage data set. This should normally be zero. If it is not, inspect the CICS and VSAM messages to determine the cause.

Terminal control

This is the DFHSTUP listing for terminal statistics.

There are a number of ways in which terminal statistics are important for performance analysis. From them, you can get the number of inputs and outputs, that is, the loading of the system by end users. Line-transmission faults and transaction faults are shown (these both have a negative influence on performance behavior).

Terminal control: resource statistics

These statistics are gathered for each terminal, including ISC, IRC and MRO sessions.

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS TERMINAL command, and are mapped by the DFHA06DS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

In addition to this, this DSECT should be used to map the terminal totals record.

Table 117. Terminal control: resource statistics

DFHSTUP name	Field name	Description
Line Id (TCAM and BSAM only)	A06TETI	is the line number for TCAM and BSAM (sequential device support) lines. The line ID is blank for all other access methods. Reset characteristic: not reset

Table 117. Terminal control: resource statistics (continued)

DFHSTUP name	Field name	Description
Term Id	A06TETI	is the identifier of each terminal as stated in the TERMINAL attribute in CEDA or in the TRMIDNT= operand in the TCT. Reset characteristic: not reset
LUname	A06LUNAM	is the terminal LU name Reset characteristic: not reset
The remainder of the information should be used for tracking terminal activity.		
Polls (TCAM and BSAM only)	A06LENP	is the number of polls that have been sent to the terminal. This field is for TCAM and BSAM only. Reset characteristic: reset to zero
Terminal Type	A06TETT	is the terminal type as defined in the TCT. For information about terminal types and their codes, see the DFHTCTTE DSECT. Reset characteristic: not reset
Acc Meth	A06EAMIB	is the terminal access method as defined in the TCT. For information about access methods and their codes, see the DFHTCTTE DSECT. Reset characteristic: not reset
Conn ID	A06SYSID	is the owning connection name of this terminal/session. Reset characteristic: not reset
No. of Xactions	A06TEOT	is the number of transactions, both nonconversational and pseudoconversational, that were started at this terminal. The transaction count is less than input messages if conversational transactions are being used. Reset characteristic: reset to zero When the operator signs off, the transaction count is not reset. At this time, message DFHSN1200 is issued containing the transaction count for that operator.
Xaction Errors	A06TEOE	is the number of transactions associated with this particular terminal that could not be started. This could mean that a transaction identifier has not been defined in the CSD data set, or that the operator does not have the proper security to enter the transaction, or that the transaction has been disabled. Reset characteristic: reset to zero When the operator signs off, the transaction error count is not reset. At this time, message DFHSN1200 is issued containing the transaction error count for that operator.

Table 117. Terminal control: resource statistics (continued)

DFHSTUP name	Field name	Description
Storage Viols	A06CSVC	is the number of storage violations that have occurred on this terminal. Reset characteristic: reset to zero
Input Messages	A06TENI	See note. Reset characteristic: reset to zero
Output Messages	A06TENO	See note. Reset characteristic: reset to zero

Note: Input messages (A06TENI) and output messages (A06TENO) are the amount of message activity per terminal. Input and output messages should represent the message traffic between CICS and the terminal. Input traffic should be the result of operator initiated input: that is, initial transaction input or input as a result of a conversational read to the terminal. Output messages should be output written by the application program or messages sent by CICS.

Input and output messages can vary because of differences in the application program being used on different terminals. ATI-initiated transactions would typically not have terminal input but could result in one or many output messages. A batch oriented terminal could initiate a single transaction that did multiple reads to the terminal resulting in multiple input messages. The differences between the remote and local terminal counts may be a result of different applications that run on them. Otherwise, they should be similar.

Xmission Errors	A06TETE	is the number of errors for this terminal, or the number of disconnects for this session. Reset characteristic: reset to zero
Pipeline messages		
NOT IN THE DFHSTUP REPORT	A06TCNT	is the total throwaway count. Reset characteristic: reset to zero
NOT IN THE DFHSTUP REPORT	A06SCNT	is the number of consecutive throwaways. Reset characteristic: reset to zero
NOT IN THE DFHSTUP REPORT	A06MCNT	is the maximum throwaway count. Reset characteristic: reset to zero
NOT IN THE DFHSTUP REPORT	A06PRTY	is the terminal priority Reset characteristic: not reset
TIOA Storage	A06STG	is the TIOA storage allowed at this terminal. Reset characteristic: reset to zero
Autoinstall times		
–Logon	A06ONTM	is time at which this terminal/session was autoinstalled. This time is expressed as *hours:minutes:seconds.decimals*. The DSECT field contains the value as a store clock (STCK) value in local time. Reset characteristic: not reset

Table 117. Terminal control: resource statistics (continued)

DFHSTUP name	Field name	Description
–Logoff	A06OFFTM	is the time at which this terminal/session was logged off. This time is expressed as *hours:minutes:seconds.decimals*. The DSECT field contains the value as a store clock (STCK) value in local time. Note that this field is only set on an Unsolicited Statistics (USS) record. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	A06GONTM	is the time at which this terminal/session was autoinstalled. The DSECT field contains the value as a store clock (STCK) value in GMT. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	A06GOFTM	is the time at which this terminal/session was logged off. The DSECT field contains the value as a store clock (STCK) value in GMT. Note that this field is only set on an Unsolicited Statistics (USS) record. Reset characteristic: not reset

Terminal control: summary resource statistics

Summary statistics are not available online.

Table 118. Terminal control: summary resource statistics

DFHSTUP name	Description
Line Id (TCAM and BSAM only)	is the line number for TCAM and BSAM (sequential device support) lines. The line ID is blank for all other access methods.
Term Id	is the identifier of each terminal as stated in the TERMINAL attribute in CEDA or in the TRMIDNT= operand in the TCT.
LUname	is the terminal LU name
The remainder of the information should be used for tracking terminal activity.	
Polls (TCAM and BSAM only)	is the total number of polls that have been sent to the terminal. This field is for TCAM and BSAM only.
Terminal Type	is the terminal type as defined in the TCT. For information about terminal types and their codes, see the DFHTCTTE DSECT.
Acc Meth	is the terminal access method as defined in the TCT. For information about access methods and their codes, see the DFHTCTTE DSECT.
Conn ID	is the last value found for the owning connection name for this terminal/session.
No. of Xactions	is the number of transactions, both nonconversational and pseudoconversational, that were started at this terminal. The transaction count is less than input messages if conversational transactions are being used. When the operator signs off, the transaction count is not reset. At this time, message DFHSN1200 is issued containing the transaction count for that operator.

Table 118. Terminal control: summary resource statistics (continued)

DFHSTUP name	Description
Xaction Errors	is the number of transactions associated with this particular terminal that could not be started. This could mean that a transaction identifier has not been defined in the CSD data set, or that the operator does not have the proper security to enter the transaction, or that the transaction has been disabled. When the operator signs off, the transaction error count is not reset. At this time, message DFHSN1200 is issued containing the transaction error count for that operator.
Storage Viols	is the number of storage violations that have occurred on this terminal.
Input Messages	See note.
Output Messages	See note.

Note: Input messages (A06TENI) and output messages (A06TENO) are the amount of message activity per terminal. Input and output messages should represent the message traffic between CICS and the terminal. Input traffic should be the result of operator initiated input: that is, initial transaction input or input as a result of a conversational read to the terminal. Output messages should be output written by the application program or messages sent by CICS.

Input and output messages can vary because of differences in the application program being used on different terminals. ATI-initiated transactions would typically not have terminal input but could result in one or many output messages. A batch oriented terminal could initiate a single transaction that did multiple reads to the terminal resulting in multiple input messages. The differences between the remote and local terminal counts may be a result of different applications that run on them. Otherwise, they should be similar.

Xmission Errors	is the number of errors for this terminal, or the number of disconnects for this session.

Transaction class (TCLASS)

Transaction class: resource statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS TRANCLASS command, and are mapped by the DFHXMCDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 119. Transaction class: resource statistics

DFHSTUP name	Field name	Description
Tclass Name	XMCTCL	is the 8–character name of the transaction class. Reset characteristic: not reset
Number Trandfs	XMCITD	is the number of installed transaction definitions that are defined to belong to this transaction class. **Note:** This will be a reference count from the latest version of the transaction definition table. This statistic is useful to identify redundant tclasses. Reset characteristic: not reset

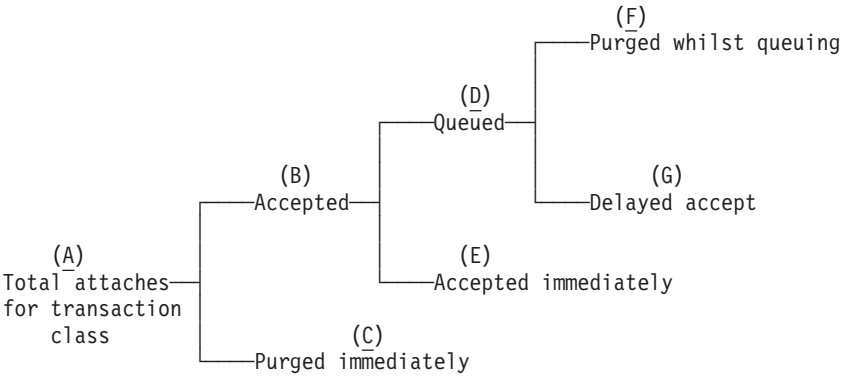
Table 119. Transaction class: resource statistics (continued)

DFHSTUP name	Field name	Description
Max Act	XMCMXT	is the maximum number of transactions in the named transaction class that may be active concurrently. Reset characteristic: not reset
Purge Thresh	XMCTH	is the queue limit of the purge threshold at which transactions in the named transaction class is purged instead of being added to the queue of transactions that are waiting for membership of the transaction class. Reset characteristic: not reset
TOTAL		
–Attaches	XMCTAT	is the total number of attach requests made for transactions in this transaction class. Reset characteristic: reset to zero
–AcptImm	XMCAI	is the number of transactions that did not have to queue to become active in this transaction class. They are accepted immediately. Reset characteristic: reset to zero
–PrgImm	XMCPI	is the number of transactions that were purged immediately because the queue reached the purge threshold for this transaction class. Reset characteristic: reset to zero
–Queued	XMCTQ	is the total number of transaction that have queued for this transaction class. Reset characteristic: reset to zero
NOT IN THE DFHSTUP REPORT	XMCAAQ	is the number of transactions that have become active in this transaction class but queued first. Reset characteristic: reset to zero
–PrgQ'd	XMCPWQ	is the number of transactions that have been purged whilst queuing for acceptance into the transaction class. This includes those transactions purged explicitly through Master Terminal, or implicitly through the purge threshold of the transaction class being lowered. Reset characteristic: reset to zero
–Q-Time	XMCTQTME	is the total time in STCK units spent waiting by those transactions that were queued in the transaction class. **Note:** This time only includes the time spent by those that have finished queuing. In order to calculate the average queuing time, current queue must be subtracted from the 'queued' count. Reset characteristic: reset to zero

Table 119. Transaction class: resource statistics (continued)

DFHSTUP name	Field name	Description
Peak Act	XMCPAT	is the highest number of active transactions reached in the transaction class. Reset characteristic: reset to current value
Peak Queued	XMCPQT	is the highest number of transactions queued waiting for admittance to the transaction class. Reset characteristic: reset to current value
Times MaxAct	XMCTAMA	is the number of separate times that the number of active transactions in the transaction class was equal to the maximum value (XMCMXT). Also registers times when maxactive setting of the tclass is zero and there are no active transactions in the tclass. Reset characteristic: reset to zero or one if transaction class is currently at its maxactive limit.
Times PrgThr	XMCTAPT	is the number of separate times that the purge threshold of the transaction class has been reached (times at purge threshold). Reset characteristic: reset to zero or one if transaction class is currently at its purge threshold limit.
CURRENT		
–Act	XMCCAT	is the current number of transactions currently active in this transaction class. Reset characteristic: not reset
–Queued	XMCCQT	is the number of transactions that are currently queuing in this transaction class. Reset characteristic: not reset
–Queue Time	XMCCQTME	is the total time in STCK units spent waiting by those transactions that are currently queuing in this transaction class. Reset characteristic: not reset

Figure 33 illustrates the transaction class statistics.




```
Attaches for Transaction class  = A                    (XMCTAT)
Accepted                        = B       (A - C)
Purged immediately              = C                    (XMCPI)
Queued                          = D       (B - E)
Accepted immediately            = E       (B - D)      (XMCAI)
Purged whilst queuing           = F                    (XMCPWQ)
Accepted after queuing          = G       (D - F)      (XMCAAQ)
```

Figure 33. The transaction class statistics

Transaction class: summary resource statistics

Summary statistics are not available online.

Table 120. Transaction class: summary resource statistics

DFHSTUP name	Description
Tclass Name	is the 8 character name of the transaction class.
Max Act	The maximum number of transactions in the named tclass that may be active concurrently.
Purge Thresh	The queue limit at which transactions in the named tclass will be purged instead of being added to the queue of transactions that are waiting for membership of the transaction class.
Total	
–Attaches	is the total number of attach requests made for transactions in this transaction class.
–AccptImm	The total number of transactions that did not have to queue to become active in this transaction class.
–PurgdImm	The total number of transactions that were purged immediately because they made the queue reach the purge threshold for this transaction class.
–Queued	The total number of transactions that have been made to queue in this transaction class.
–PurgQ'd	The total number of transactions that have been purged whilst queuing for acceptance into the transaction class. This includes those transactions purged explicitly via Master Terminal, or implicitly via the purge threshold of the transaction class being lowered.
–Queuing-Time	The total time spent waiting by those transactions that were queued. Note this time only includes the time spent by those have finished queuing. In order to calculate the average queuing time, current queue must be subtracted from the 'queued' count.
Peak Act	The total highest number of active transactions reached in the transaction class.
Peak Queued	The total highest number of transactions queued waiting for admittance to the transaction class.

Table 120. Transaction class: summary resource statistics (continued)

DFHSTUP name	Description
Times Max Act	The total number of separate times that the number of active transactions in the transaction class was equal to the maximum value.
Times PurgeThr	The total number of separate times that the purge threshold has been reached.
Average Queuing-Time	The average time spent waiting by those transactions that were queued.

Transaction manager

Transaction manager: global statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS TRANSACTION command, and are mapped by the DFHXMGDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 121. Transaction manager: global statistics

DFHSTUP name	Field name	Description
Total number of transactions (user + system)	XMGNUM	is the number of transactions (user + system) that have run in the system.
Current MAXTASKS limit	XMGMXT	Reset characteristic: reset to zero is the latest MXT value (expressed as a number of tasks) specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM MAXTASKS(value) or EXEC CICS SET SYSTEM MAXTASKS(fullword binary data-value) commands.
Current number of active user transactions	XMGCAT	Reset characteristic: not reset is the current number of active user transactions in the system.
Current number of MAXTASK queued user transactions	XMGCQT	Reset characteristic: not reset is the current number of queued user transactions in the system. Note that this does not include transactions queueing for transaction class membership. Note that the current queueing time for these transactions is in field XMGCQTME.
Times the MAXTASKS limit reached	XMGTAMXT	Reset characteristic: not reset is the number of times the MXT limit has been reached
Peak number of MAXTASK queued user transactions	XMGPQT	Reset characteristic: reset to zero (or one if at MXT) is the peak number of MAXTASK queued user transactions reached in the system. Reset characteristic: reset to current value (XMGCAT)

Table 121. Transaction manager: global statistics (continued)

DFHSTUP name	Field name	Description
Peak number of active user transactions	XMGPAT	is the number of user transactions that have become active. Reset characteristic: reset to zero
Total number of active user transactions	XMGTAT	is the total number of user transactions that have become active. Reset characteristic: reset to zero
Number of MAXTASK delayed user transactions	XMGTDT	is the number of user transactions that had to queue for MXT reasons. This value does not include those transactions that are currently queueing for MXT (see XMGCQT). Note that the queueing time for these transactions is in field XMGTQTME. Reset characteristic: reset to zero
Total MAXTASK queuing time	XMGTQTME	is the total time spent waiting by those user transactions that had to queue for MXT reasons. This value does not include the time spent by those transactions that are currently queueing for MXT (see XMGCQTME). Reset characteristic: reset to zero
Total MAXTASK queuing time of currently queued user transactions	XMGCQTME	is the total time spent waiting so far by those user transactions currently queuing for MXT reasons. Reset characteristic: reset to zero
NOT IN THE DFHSTUP REPORT	XMGTNUM	is the total of user and system transactions attached to date, up to the time of the last statistics reset. **Note:** The total of XMGNUM and XMGTNUM represents the total number of transactions attached so far. Reset characteristic: not reset Reset characteristic: reset to XMGNUM + XMGTNUM at the time of the last reset.

Transaction manager: summary global statistics

Summary statistics are not available online.

Table 122. Transaction manager: summary global statistics

DFHSTUP name	Description
Total number of transactions (user and system)	is the total number of tasks that have run in the system.
MAXTASK limit	is the last MXT value (expressed as a number of tasks) that was specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM MAXTASKS(value) or EXEC CICS SET SYSTEM MAXTASKS(fullword binary data-value) commands.
Times the MAXTASK limit reached	is the total number of times MXT has been reached.
Peak number of active user transactions	is the peak number of active user transactions reached in the system.
Total number of active user transactions	is the total number of user transactions that have become active.

Table 122. Transaction manager: summary global statistics (continued)

DFHSTUP name	Description
Total number of MAXTASK delayed user transactions	is the total number of transactions that had to queue for MXT reasons.
Total MAXTASK queuing time	is the total time spent waiting by those user transactions that had to queue for MXT reasons.
Average MAXTASK queuing time of queued transactions	is the average time spent waiting by those user transactions that had to queue for MXT reasons.

Transaction manager: resource statistics

There are two sections in the DFHSTUP report for transaction manager (resource) statistics:

- Transaction: Resource Information (see Transaction statistics: resource information)
- Transaction: Integrity Information (see "Transactions: integrity information" on page 480)

Transaction statistics: resource information

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS command and are mapped by the DFHXMRDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the command, see the *CICS System Programming Reference* manual.

The transaction statistics show how often each transaction is called.

Table 123. Transaction statistics: resource information

DFHSTUP name	Field name	Description
Trans ID	XMRTI	is the transaction identifier associated with the transaction definition.
Program Name	XMRPN	Reset characteristic: not reset is the name of the initial program to which the transaction linked.
Tclass Name	XMRTCL	Reset characteristic: not reset is the name of the transaction class in which the transaction is defined.
Prty	XMRPRTY	Reset characteristic: not reset is the priority of the transaction, from 0–255.
Remote Name	XMRRNAM	Reset characteristic: not reset is the name of the transaction on the remote system.
Remote Sysid	XMRRSYS	Reset characteristic: not reset is the name of the remote system where the transaction resides.
Dynamic	XMRDYN	Reset characteristic: not reset indicates whether the transaction has been defined as DYNAMIC=YES (Y) or DYNAMIC=NO (N). Reset characteristic: not reset

Table 123. Transaction statistics: resource information (continued)

DFHSTUP name	Field name	Description
Attach Count	XMRAC	is the number of times that this transaction has been attached, Reset characteristic: reset to zero
Retry Count	XMRRC	is the number of times that this transaction definition has been used to retry a transaction.
Dynamic Local	XMRDLC	Reset characteristic: reset to zero is the number of times the dynamic transaction routing exit has chosen to run this transaction on the local system. This field is zero if the transaction was not defined as DYNAMIC=YES. For further information about dynamic transaction routing, see the programming information in the *CICS Customization Guide*.
Dynamic Remote	XMRDRC	Reset characteristic: reset to zero is the number of times the dynamic transaction routing exit has chosen to run this transaction on a remote system. This field is zero if the transaction was not defined as DYNAMIC=YES. For further guidance about dynamic transaction routing, see the programming information in the *CICS Customization Guide*.
Remote Starts	XMRRSC	Reset characteristic: reset to zero is the number of attempts to start this transaction on a remote system. This may not necessarily be the same as the number of successful starts.
Storage Violations	XMRSVC	Reset characteristic: reset to zero is the number of storage violations for this transaction that have been detected by CICS storage management. This is a serious concern if it occurs in a production system. You should act immediately to identify the cause of the problem because it can lead to data corruption, and therefore should not be allowed to continue in an operational system. Reset characteristic: reset to zero

Transactions: summary resource statistics

Summary statistics are not available online.

Table 124. Transactions: summary resource information

DFHSTUP name	Description
Trans ID	is the transaction identifier associated with the transaction definition.

Table 124. Transactions: summary resource information (continued)

DFHSTUP name	Description
Program Name	is the name of the initial program to which the transaction was linked.
Tclass Name	is the name of the transaction class in which the transaction is defined.
Prty	is the priority of the transaction, from 1–255.
Remote Name	is the name of the transaction on the remote system.
Remote Sysid	is the name of the remote system where the transaction resides.
Dynamic	indicates whether the transaction has been defined as DYNAMIC=YES (Y) or DYNAMIC=NO (NO).
Attach Count	is the total number of times this transaction has been attached.
Retry Count	is the total number of times that this transaction definition has been used to retry a transaction.
Dynamic Local	is the total number of times the dynamic transaction routing exit has chosen to run this transaction on the local system. This field is zero if the transaction was not defined as DYNAMIC=YES. For further guidance information about dynamic transaction routing. For programming information about dynamic transaction routing, see the *CICS Customization Guide*.
Dynamic Remote	is the total number of times the dynamic transaction routing exit has chosen to run this transaction on a remote system. This field is zero if the transaction was not defined as DYNAMIC=YES. For further information about dynamic transaction routing, see the *CICS Customization Guide*.
Remote Starts	is the total number of times this transaction definition has been used to start a transaction remotely.
Storage Violations	is the total number of storage violations for this transaction that have been detected by CICS storage management. This is a serious concern if it occurs in a production system. You should act immediately to identify the cause of the problem because it can lead to data corruption, and therefore should not be allowed to continue in an operational system.

Transactions: integrity information

The integrity information statistics show the potential integrity exposures that may have occurred during transaction execution as a result of inabilities to shunt UOWs, or forcing of shunted UOWs to complete regardless of the decisions made by participating systems.

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS command and are mapped by the DFHXMRDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS, see the *CICS System Programming Reference* manual.

Table 125. Transaction statistics: integrity information

DFHSTUP name	Field name	Description
Trans ID	XMRTI	is the transaction identifier associated with the transaction definition. Reset characteristic: not reset

Table 125. Transaction statistics: integrity information (continued)

DFHSTUP name	Field name	Description
Indoubt Wait	XMRIWTOP	Is the indicator of whether the transaction has been defined to support Indoubt Waiting in the event of an two-phase commit indoubt window failure. This means the failing UOW will be shunted by the CICS recovery manager awaiting resynchronisation with its coordinator. The indoubt wait option can have the following settings: • XMRIWTY = 'Y' = Transaction can support waiting • XMRIWTN = 'N' = Transaction cannot support waiting. Reset characteristic: not reset
Indoubt Wait timeout	XMRITOV	Is the indoubt wait timeout limit defined for this transaction, specified in minutes. This value has meaning only if the transaction is also defined to be able to wait indoubt (see XMRIWTOP). A value of zero, specifies that there is no timeout should this transaction be shunted by the CICS recovery manager. Reset characteristic: not reset
Indoubt Action	XMRIACTN	Is an indicator of which way this transaction will commit its UOWs in the event of not being able to wait indoubt (shunted), when an indoubt wait failure occurs. Or if the transaction had been waiting that, the timeout value specified has expired. Both of these events will force a resolution of the UOW in the direction specified by this field. The values can be : • XMRIACOM = 'C' = UOW will syncpoint forwards • XMRIABCK = 'B' = UOW will syncpoint backwards (rollback) Reset characteristic: not reset
Indoubt Waits	XMRIWAIT	Is the number of indoubt waits (shunts) that have occurred for UOWs executing on behalf of this transaction. Reset characteristic: not reset

Indoubt action forced

The following set of statistics are further breakdowns of the recovery manager global statistics to aid further isolation of potential integrity exposures.

Table 125. Transaction statistics: integrity information (continued)

DFHSTUP name	Field name	Description
–Trandefn	XMRFATXN	Is the number of times this transaction id had a UOW that could not be shunted when an indoubt failure occurred, because the transaction definition for this transaction id specified that it could not support indoubt waiting (ie. XMRIWTOP = XMTIWTN). The UOW would have been forced to resolve in the direction specified by XMRIACTN, regardless of the actions taken by any other participating region in this distributed UOW. Reset characteristic: not reset
–Timeout	XMRFAIT	Is the number of times this transaction id had a UOW that, although shunted because of an indoubt failure, had the wait for resynchronization with its recovery coordinator terminated prematurely, because the indoubt wait timeout value (XMRITOV) had been exceeded. The UOW would have been forced to resolve in the direction specified by XMRIACTN, regardless of the actions taken by any other participating region in this distributed UOW. Reset characteristic: not reset
–Operator	XMRFAOP	Is the number of times this transaction id had a UOW that although shunted because of an indoubt failure, had the wait for resynchronization with its recovery coordinator terminated prematurely, because an operator (CEMT) or SPI command forced a resolution. The UOW would have been forced to resolve in the direction specified by XMRIACTN by default, or in the direction specified by the operator, regardless of the actions taken by any other participating region in this distributed UOW. Reset characteristic: reset to zero
–No waiting	XMRFANW	Is the number of times this transaction id had a UOW that could not be shunted when an indoubt failure occurred, although the transaction definition specified that it could (XMRIWTOP = XMRIWTY), because the resource managers (RMIs) or CICS resources or CICS connections used by the UOW could not support indoubt waiting (shunting). The UOW would have been forced to resolve in the direction specified by XMRIACTN, regardless of the actions taken by any other participating region in this distributed UOW. Reset characteristic: reset to zero

Table 125. Transaction statistics: integrity information (continued)

DFHSTUP name	Field name	Description
–Other	XMRFAOT	Is the number of times this transaction id had a UOW that although shunted because of an indoubt failure, had the wait for resynchronization with its recovery coordinator terminated prematurely, for reasons other than those stated above. This could be a cold started recovery coordinator, a resynchronization protocol violation or failure, or level of resource manager (RMI) adaptor changes etc. The UOW would have been forced to resolve in the direction specified by XMRIACTN by default, or in the direction specified by the operator, regardless of the actions taken by any other participating region in this distributed UOW. Reset characteristic: reset to zero
Action mismatch	XMRAMISM	is the number of times this transaction id had a UOW that was forced to resolve using the indoubt action attribute, whether by definition, option or operator override (as detailed in the above fields), and on doing so detected an indoubt action attribute mismatch with a participating system or resource manager (RMI). For example, a participating system in a distributed UOW resolves its work forward while other systems back out theirs. The opposite also applies. Reset characteristic: reset to zero

Transactions: summary integrity information

Summary statistics are not available online.

Table 126. Transactions: summary integrity information

DFHSTUP name	Description
Trans ID	is the transaction identifier associated with the transaction definition.
Indoubt Wait	is the last value encountered for the indicator of whether the transaction has been defined to support indoubt waiting in the event of an two-phase commit indoubt window failure. This means the failing UOW will be shunted by the CICS recovery manager awaiting resynchronization with its coordinator.
Indoubt Wait timeout	is the last value encountered for the indoubt wait timeout limit defined for this transaction, specified in minutes. This value only has any meaning if the transaction is also defined to be able to wait indoubt (see XMRIWTOP). A value of zero specifies that there is no timeout should this transaction be shunted by the CICS recovery manager.
Indoubt Action	is the last value encountered for the indicator of which way this transaction will commit its UOWs in the event of not being able to wait indoubt (shunted), when an indoubt wait failure occurs. Or if the transaction had been waiting, that the timeout value specified had expired. Both of these events will force a resolution of the UOW in the direction specified by this field.

Table 126. Transactions: summary integrity information (continued)

DFHSTUP name	Description
Indoubt Waits	is the number of indoubt waits (shunts) that have occurred for UOWs executing on behalf of this transaction.
Indoubt action forced	
–Trandefn	is the number of times this transaction id had a UOW that could not be shunted when an indoubt failure occurred, because the transaction definition for this transaction id specified that it could not support indoubt waiting (ie. XMRIWTOP = XMTIWTN). The UOW would have been forced to resolve in the direction specified by XMRIACTN, regardless of the actions taken by any other participating region in this distributed UOW.
–Timeout	is the number of times this transaction id had a UOW that although shunted because of an indoubt failure, had the wait for resynchronization with its recovery coordinator terminated prematurely, because the indoubt wait timeout value (XMRITOV) had been exceeded. The UOW would have been forced to resolve in the direction specified by XMRIACTN, regardless of the actions taken by any other participating region in this distributed UOW.
–Operator	is the number of times this transaction id had a UOW that although shunted because of an indoubt failure, had the wait for resynchronization with its recovery coordinator terminated prematurely, because an operator (CEMT) or SPI command forced a resolution. The UOW would have been forced to resolve in the direction specified by XMRIACTN by default, or in the direction specified by the operator, regardless of the actions taken by any other participating region in this distributed UOW.
–No waiting	is the number of times this transaction id had a UOW that could not be shunted when an indoubt failure occurred, even though the transaction definition specified that it could (XMRIWTOP = XMRIWTY), because the resource managers (RMIs) or CICS resources or CICS connections used by the UOW could not support indoubt waiting (shunting). The UOW would have been forced to resolve in the direction specified by XMRIACTN, regardless of the actions taken by any other participating region in this distributed UOW.
–Other	is the number of times this transaction id had a UOW that although shunted because of an indoubt failure, had the wait for resynchronization with its recovery coordinator terminated prematurely, because reasons other than those stated above. This could be a cold started recovery coordinator, a resynchronization protocol violation or failure, or level of resource manager (RMI) adaptor changes etc. The UOW would have been forced to resolve in the direction specified by XMRIACTN by default, or in the direction specified by the operator, regardless of the actions taken by any other participating region in this distributed UOW.
Action mismatch	is the number of times this transaction id had a UOW that was forced to resolve using the indoubt action attribute, whether by definition, option or operator override (as detailed in the above fields), and on doing so detected an indoubt action attribute mismatch with a participating system or resource manager (RMI). For example, a participating system in a distributed UOW resolves its work forward while other systems back out theirs. The opposite also applies.

Transient data

Transient data: global statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS TDQUEUE command, and are mapped by the DFHTQGDS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

Table 127. Transient data: global statistics

DFHSTUP name	Field name	Description
In the statistics produced for the intrapartition data set:		
Control interval size	TQGACISZ	is the size of the control interval, expressed in bytes.
Control intervals	TQGANCIS	Reset characteristic: not reset is the number of control intervals in the intrapartition data set DFHINTRA.
Current control intervals in use	TQGACTCI	Reset characteristic: not reset is the current number of control intervals in the intrapartition data set DFHINTRA.
Peak control intervals used	TQGAMXCI	Reset characteristic: not reset is the peak value of the number of control intervals concurrently active in the system.
Times NOSPACE occurred	TQGANOSP	Reset characteristic: reset to current value is the number of times that a NOSPACE condition has occurred.
Writes to intrapartition data set	TQGACTPT	Reset characteristic: reset to zero is the number of WRITEs to the intrapartition transient data set. This includes both WRITEs needed for recovery (see below) and WRITEs forced by the buffer being needed to accommodate another CI. I/O activity caused by the latter reason can be minimized by increasing the buffer allocation.
Reads from intrapartition data set	TQGACTGT	Reset characteristic: reset to zero is the number of times a CI has to be read from disk. Increasing the buffer allocation decreases this activity.
Formatting writes	TQGACTFT	Reset characteristic: reset to zero is the number of times a new CI was written at the end of the data set in order to increase the amount of available space.
I/O errors	TQGACTIO	Reset characteristic: reset to zero is the number of input/output errors that have occurred during this run of CICS.
In the statistics produced for buffer usage:		Reset characteristic: reset to zero

Table 127. Transient data: global statistics (continued)

DFHSTUP name	Field name	Description
Intrapartition buffers	TQGANBFA	is the number of transient data buffers specified in the system initialization table (SIT) or in the SIT overrides. The number of buffers allocated may exceed the number requested. Reset characteristic: not reset
Current buffers containing valid data	TQGACNIU	is the current number of intrapartition buffers that contain valid data. Reset characteristic: not reset
Peak intra. buffers containing valid data	TQGAMXIU	is the peak number of intrapartition buffers which contain valid data. Reset characteristic: reset to current value
Intrapartition accesses	TQGATNAL	is the number of times intrapartition buffers have been accessed. Reset characteristic: reset to current value
Current concurrent buffer accesses	TQGACNAL	is the current value of the number of concurrent intrapartition buffer accesses. Reset characteristic: not reset
Peak concurrent intrapartition accesses	TQGAMXAL	is the peak value of the number of concurrent intrapartition buffer accesses. Reset characteristic: reset to current value
Intrapartition buffer waits	TQGATNWT	is the number of times a request was queued because all buffers were allocated to other tasks. A buffer wait occurs if the required control interval is already in a locked buffer, and therefore unavailable, even if there are other buffers available. Reset characteristic: reset to current value
Current intrapartition buffer waits	TQGACNWT	is the current number of requests queued because no buffers were available. Reset characteristic: not reset
Peak intrapartition buffer waits	TQGAMXWT	is the peak number of requests queued because no buffers were available. Reset characteristic: reset to current value
All of the intrapartition data set statistics above are printed, even if the values reported are zero.		
CICS produces the following statistics for multiple strings:		
Number of strings	TQGSNSTA	is the number of strings currently active.
Times string accessed	TQGSTNAL	Reset characteristic: not reset is the number of times a string was accessed. Reset characteristic: reset to current value
Current concurrent string accesses	TQGSCNAL	is the current number of strings concurrently accessed in the system. Reset characteristic: not reset

Table 127. Transient data: global statistics (continued)

DFHSTUP name	Field name	Description
Peak concurrent string accesses	TQGSMXAL	is the peak number of strings concurrently accessed in the system.
Intrapartition string waits	TQGSTNWT	Reset characteristic: reset to current value is the number of times that tasks had to wait because no strings were available.
Current intrapartition string waits	TQGSCNWT	Reset characteristic: reset to current value is the current number of concurrent string waits in the system.
Peak string waits	TQGSMXWT	Reset characteristic: not reset is the peak number of concurrent string waits in the system. Reset characteristic: reset to current value

Transient data: summary global statistics

Summary statistics are not available online.

Table 128. Transient data: summary global statistics

DFHSTUP name	Description
In the statistics produced for the intrapartition data set:	
Control interval size	is the last value encountered for the size of the control interval, expressed in bytes.
Peak control intervals used	is the peak number of control intervals concurrently in the system.
Times NOSPACE occurred	is a total number of times that a NOSPACE condition has occurred.
Writes to intrapartition data set	is the total number of WRITEs to the temporary storage data set. This includes both WRITEs needed for recovery (see below) and WRITEs forced by the buffer being needed to accommodate another CI. I/O activity caused by the latter reason can be minimized by increasing the buffer allocation.
Reads from intrapartition data set	is the total number of times a CI has to be read from disk. Increasing the buffer allocation decreases this activity.
Formatting writes	is the total number of times a new CI was written at the end of the data set in order to increase the amount of available space.
I/O errors	is the total number of input/output errors that have occurred during this run of CICS.
In the statistics produced for buffer usage:	
Intrapartition buffers	is the last value encountered for the number of transient data buffers specified by the TD system initialization parameter. The number of buffers allocated may exceed the number requested.
Peak intra. buffers containing valid data	is the peak number of intrapartition data sets which contain valid data.
Intrapartition accesses	is the total number of times that intrapartition data sets have been accessed.
Peak concurrent intrapartition accesses	is the peak number of concurrent intrapartition data set accesses.
Intrapartition buffer waits	is the total number of times a request was queued because all buffers were allocated to other tasks. A buffer wait occurs if the required control interval is already in a locked buffer, and therefore unavailable, even if there are other buffers available.
Peak intrapartition buffer waits	is the peak number of requests queued because no buffers were available.

Table 128. Transient data: summary global statistics (continued)

DFHSTUP name	Description
In the statistics produced for the intrapartition data set: All of the intrapartition data set statistics above are printed, even if the values reported are zero.	
CICS produces the following statistics for multiple strings:	
Times strings accessed	is the total number of times a string was accessed.
Peak concurrent string accesses	is the peak number of strings concurrently accessed in the system.
Intrapartition string waits	is the total number of times that tasks had to wait because no strings were available.
Peak string waits	is the peak number of concurrent string waits in the system.

Transient data: resource statistics

These statistics are collected for each queue. You can use the information from the statistics for each queue to calculate the average number of transient data accesses per transaction. The items in this listing reflect the information you placed in the definition for the transient data queue. The statistics are available online, and are mapped by the DFHTQRDS DSECT.

The TQRQTYPE field is not displayed in the DFHSTUP report. It signifies the queue type, which can be one of:

- TQRQTEXT (X'01') for extrapartition queues
- TQRQTINT (X'02') for intrapartition queues
- TQRQTIND (X'03') for indirect queues
- TQRQTREM (X'04') for remote queues.

TQRQTYPE is reset to zero.

Resource statistics for intrapartition transient data queues

Table 129. Transient data: resource statistics

DFHSTUP name	Field name	Description
Queue id	TQRQID	is the destination identifier (queue) that you specified in transient data queue definition. Reset characteristic: Not reset
Request Counts		
–Number of Writes	TQRWRITE	is the total number of write operations. Reset characteristic: Reset to zero
–Number of Reads	TQRREAD	is the total number of read operations. Reset characteristic: Reset to zero
–Number of Deletes	TQRDELET	is the total number of delete requests. Reset characteristic: Reset to zero
ATI Information		
–Triggerlevel	TQRTRIGL	is the value of the ATI trigger level. If the number of items in this queue reaches this value the transaction id in TQRATRAN is attached to process the items in the queue. Reset characteristic: Not reset

Table 129. Transient data: resource statistics (continued)

DFHSTUP name	Field name	Description
–Tran Id	TQRATRAN	is the id of the transaction that will be scheduled against a terminal/session or in the background (see TQRFTYPE) when the trigger level (TQRTRIGL) has been reached. Reset characteristic: Not reset
–Facility Type	TQRFTYPE	is the ATI facility type for this transient data queue. This will be where and how the transaction id in TQRATRAN is attached when the ATI trigger level (TQRTRIGL) is reached. It can have the following values:- • TQRFTNA X'00' Not Applicable • TQRFTTRM X'01' Terminal • TQRFTSYS X'02' System • TQRFTNTE X'03' No terminal. Reset characteristic: Not reset
–Facility Name	TQRFNAME	is the id of the system or terminal that the trigger transaction will be attached against. This value is blank when there is no facility. Reset characteristic: Not reset
–No. of triggers	TQRTRIGN	is the number of times the trigger transaction (TQRATRAN) has been scheduled, as a result of the trigger level (TQRTRIGL) being exceeded. Reset characteristic: Reset to zero
Recovery		
–Rcvy type	TQRRTYPE	is the recoverable type of this transient data queue. It can have the following values:- • TQRRTNA X'00' Not applicable • TQRRTPH X'01' Physical recoverable • TQRRTLG X'02' Logical recoverable • TQRRTNR X'03' Non-recoverable Reset characteristic: Not reset

Table 129. Transient data: resource statistics (continued)

DFHSTUP name	Field name	Description
–Wait opt.	TQRWAIT	is an indicator of whether any transactions that use this queue will be able, in the event of losing the connection to their recovery coordinator, to wait indoubt (shunted). If the queue supports indoubt waiting (TQRWTYES) then the locks that are associated with that UOW will be held until syncpoint resolution. If not, the UOW will be committed (forward or backward) at the time of indoubt failure according to the settings in the transaction definition and the locks released as a result. This field has meaning only if the queue is logically recoverable. The indoubt wait option can have the following settings: • TQRWTNA X'00' Not Applicable • TQRWTYES X'01' Queue supports indoubt waiting • TQRWTNO X'02' Does not support indoubt waiting Reset characteristic: Not reset
–Wait Action	TQRWAITA	is an indicator of whether this transient data queue will reject or suspend subsequent requests to this queue. This can be when a UOW that has used this queue has been shunted because of an indoubt failure and is therefore retaining enqueues against this queue. This field has no meaning if the queue is non-recoverable or does not support indoubt waiting (see TQRWAIT). The possible values for this field are: • TQRWANA X'00' Not Applicable • TQRWAREJ X'01' Further requests will be rejected • TQRWAQUE X'02' Further requests will be queued Reset characteristic: Not reset
DFHINTRA usage		
–Current CIs used	TQRCCIUS	is the number of Control intervals (CIs) that are currently in use on the DFHINTRA dataset by this queue. Reset characteristic: Not reset
–Peak CIs used	TQRPCIUS	is the peak number of Control intervals (CIs) that have been used on the DFHINTRA dataset by this queue. Reset characteristic: Reset to current
–Current items	TQRCNITM	is the current number of items in this intrapartition queue. Reset characteristic: Not reset

Resource statistics for extrapartition transient data queues

Table 130. Transient data: resource statistics

DFHSTUP name	Field name	Description
Queue ID	TQRQID	is the destination identifier (queue) that you specified in transient data queue definition.
DD name (assoc.)	TQRDDNM	Reset characteristic: Not reset is the associated DD name of this dataset in the CICS start-up JCL, or as defined by under CEDA.
Dataset name (Destination/origin of data)	TQRDSNNM	Reset characteristic: Not reset is the data set name of the extrapartition transient data queue.
Member Name	TQRPDSMN	Reset characteristic: Not reset is the name of a member in the partitioned data set referenced by the ddname for the extrapartition transient data queue.
I/O Type	TQRIOTYP	Reset characteristic: Not reset is an indicator of the input/output type of the extrapartition dataset. It may contain one of the following values:- • TQRIONA X'00' Not Applicable • TQRIOIN X'01' Input • TQRIOOUT X'02' Output • TQRIORDB X'03' Readback (input but read backwards)
No. of Writes	TQRWRITE	Reset characteristic: Not reset is the total number of write requests.
No. of Reads	TQRREAD	Reset characteristic: Reset to zero is the total number of read requests. Reset characteristic: Reset to zero

Resource statistics for indirect transient data queues

Table 131. Transient data: resource statistics

DFHSTUP name	Field name	Description
Queue ID	TQRQID	is the destination identifier (queue) that you specified in transient data queue definition.
Indirect Queue id	TQRIQID	Reset characteristic: Not reset is the name of the indirect queue.
Request Counts		Reset characteristic: Not reset
–Writes	TQRWRITE	is the total number of write requests.
–Reads	TQRREAD	Reset characteristic: Reset to zero is the total number of read requests. Reset characteristic: Reset to zero

Table 131. Transient data: resource statistics (continued)

DFHSTUP name	Field name	Description
–Deletes	TQRDELET	is the total number of delete requests. Reset characteristic: Reset to zero

Resource statistics for remote transient data queues

Table 132. Transient data: resource statistics

DFHSTUP name	Field name	Description
Queue Id	TQRQID	is the destination identifier (queue) that you specified in transient data queue definition. Reset characteristic: Not reset
Remote		
–Queue	TQRRQID	is the name of the queue on the remote system (TQRRSYS). Reset characteristic: Not reset
–Sysid	TQRRSYS	is the connection id of the CICS system that actually owns this queue. Reset characteristic: Not reset
Request Counts		
–Writes	TQRWRITE	is the total number of write requests. Reset characteristic: Reset to zero
–Reads	TQRREAD	is the total number of read requests. Reset characteristic: Reset to zero
–Deletes	TQRDELET	is the total number of delete requests, Reset characteristic: Reset to zero

Transient data: summary resource statistics

Summary statistics are not available online.

Table 133. Transient data: summary resource statistics

DFHSTUP name	Description
Queue ID	is the destination identifier (queue) that you specified in transient data queue definition.
Request Counts	
–Number of Writes	is the total number of write operations.
–Number of Reads	is the total number of read operations.
–Number of Deletes	is the total number of delete requests.
ATI Information	
–Triggerlevel	is the ATI trigger level
–Tran Id	is the ATI transaction id
–Facility Type	is the ATI facility type
–Facility Name	is the ATI facility name
–No. of triggers	is the number of times a trigger transaction has been attached.
Recovery	
–Rcvy type	is the recovery type
–Wait opt.	Indoubt waiting supported
–Wait Action	Indoubt action
DFHINTRA usage	

Table 133. Transient data: summary resource statistics (continued)

DFHSTUP name	Description
–Current CIs used	is the current number of CIs used by this intrapartition queue
–Peak CIs used	is the peak number of CIs used by this intrapartition queue
–Current items	is the current number of items in this intrapartition queue

Table 134. Transient data: summary resource statistics

DFHSTUP name	Description
Queue ID	is the destination identifier (queue) that you specified in transient data queue definition.
DDNAME (assoc.)	is the DDNAME of the extrapartition queue.
Dataset name (Destination/origin of data)	is the data set name of the extrapartition queue.
I/O Type	is the type of I/O data set. Can be one of input, output or readback.
Member Name	is the name of a member in the partitioned data referenced by the ddname for the extrapartition transient data queue.
No. of Writes	is the total number of write operations to the output data set.
No. of Reads	is the total number of read operations from the output data set.

Table 135. Transient data: summary resource statistics

DFHSTUP name	Description
Queue ID	is the destination identifier (queue) that you specified in transient data queue definition.
Indirect Queue id	is the name of the indirect queue.
Request Counts	
–Writes	is the total number of write operations to the output data set.
–Reads	is the total number of read operations from the output data set.
–Deletes	is the total number of queues deleted.

Table 136. Transient data: summary resource statistics

DFHSTUP name	Description
Queue Id	is the destination identifier (queue) that you specified in transient data queue definition.
Remote	
–Queue	is the name of the remote queue.
–Sysid	is the name of the remote system.
Request Counts	
–Writes	is the total number of write operations to the output data set.
–Reads	is the total number of read operations from the output data set.
–Deletes	is the total number of queues deleted.

User domain statistics

These statistics are not available online, and are mapped by the DFHUSGDS DSECT.

User domain: global statistics

Table 137. User domain statistics: global statistics

DFHSTUP name	Field name	Description
Timeout mean reuse time	USGTOMRT	the average time user instances remain on the timeout queue until they are removed. Reset characteristic: reset to zero
Timeout reuse count	USGTORC	the number of times a user instance is reused while it was being timed out. Reset characteristic: reset to zero
Timeout expiry count	USGTOEC	the number of times a user instance remains on the timeout queue for a full USRDELAY interval without being reused, and is deleted. Reset characteristic: reset to zero
Directory reuse count	USGDRRC	the number of times a directory entry was reused. Reset characteristic: reset to zero
Directory not found count	USGDRNFC	the number of times a user instance was not found in the directory, but was later successfully added. Reset characteristic: reset to zero

User domain: Summary global statistics

Summary statistics are not available online.

Table 138. User Domain: summary global statistics

DFHSTUP name	Description
Average timeout reuse time	is the average time user instances remain on the timeout queue until they are removed.
Timeout reuse count	is the number of times a user instance is reused while being timed out.
Timeout expiry count	is the number of times a user instance remains on the timeout queue for a full USRDELAY interval without being reused.
Directory reuse count	records how many times an existing user interface is reused.
Directory not found count	records the number of times the user instance needs to be added if it doesnot already exist in the directory.

VTAM statistics

These statistics can be accessed online using the EXEC CICS COLLECT STATISTICS VTAM command, and are mapped by the DFHA03DS DSECT. For programming information about the EXEC CICS COLLECT STATISTICS command, see the *CICS System Programming Reference* manual.

VTAM: global statistics

Table 139. VTAM statistics: global statistics

DFHSTUP name	Field name	Description
Times at RPL maximum	A03RPLXT	is the number of times the peak RPL posted value (A03RPLX) was reached. Reset characteristic: reset to zero

Table 139. VTAM statistics: global statistics (continued)

DFHSTUP name	Field name	Description
Peak RPLs posted	A03RPLX	is the maximum number of receive-any request parameter lists (RPLs) that are posted by VTAM on any one dispatch of terminal control.
Short on storage count	A03VTSOS	Reset characteristic: reset to zero is a counter that is incremented in the VTAM SYNAD exit in the CICS terminal control program each time VTAM indicates that there is a temporary VTAM storage problem.
Dynamic opens count	A03DOC	Reset characteristic: reset to zero is the number of times the VTAM access method control block (ACB) was opened through the control terminal. If VTAM is started before CICS and stays active for the whole CICS run, this value is zero.
Current LUs in session	A03LUNUM	Reset characteristic: reset to zero is the current number of LUs in session. The types of LU that are included are: • LU6.1 primaries and secondaries in session (bound) • LU6.2 primaries and secondaries in session (bound) • VTAM terminals.
HWM LUs in session	A03LUHWM	Reset characteristic: not reset. is the current highest number of LUs logged on. The types of LU that are included are: • LU6.1 primaries and secondaries in session (bound) • LU6.2 primaries and secondaries in session (bound) • VTAM terminals.
PS inquire count	A03PSIC	Reset characteristic: reset to current value. is the number of times CICS issued INQUIRE OPTCD=PERSESS.
PS nib count	A03PSNC	Reset characteristic: reset to current value. is the number of VTAM sessions that persisted.
PS opndst count	A03PSOC	Reset characteristic: reset to current value. is the number of persisting sessions that were successfully restored.
PS unbind count	A03PSUC	Reset characteristic: reset to current value. is the number of persisting sessions that were terminated.
		Reset characteristic: reset to current value.

Table 139. VTAM statistics: global statistics (continued)

DFHSTUP name	Field name	Description
PS error count	A03PSEC	is the number of persisting sessions that were already unbound when CICS tried to restore them. Reset characteristic: reset to current value.

VTAM Summary global statistics

Summary statistics are not available online.

Table 140. VTAM: summary global statistics

DFHSTUP name	Description
Times at RPL maximum	is the total number of times the maximum RPL posted value (A03RPLX) was reached.
Peak RPLs posted	is the peak number of receive-any request parameter lists (RPLs) that are posted by VTAM on any one dispatch of terminal control.
Short on storage count	is a counter that is incremented in the VTAM SYNAD exit in the CICS terminal control program each time VTAM indicates that there is a temporary VTAM storage problem.
Dynamic opens count	is the total number of times that the VTAM access method control block (ACB) was opened through the control terminal. If VTAM is started before CICS and stays active for the whole CICS run, this value is 0.
Current LUs in session	is the average value for the number of LUs logged on.
HWM LUs in session	is the highest value of the number of LUs logged on.
PS inquire count	is the total number of times CICS issued INQUIRE OPTCD=PERSESS.
PS nib count	is the total number of VTAM sessions that persisted.
PS opndst count	is the total number of persisting sessions that were successfully restored.
PS unbind count	is the total number of persisting sessions that were terminated.
PS error count	is the total number of persisting sessions that were already unbound when CICS tried to restore them.

Appendix B. Shared temporary storage queue server statistics

This appendix provides information on statistics that are obtained for the shared TS queue server.

This appendix describes the following:

- "Shared TS queue server: coupling facility statistics"
- "Shared TS queue server: buffer pool statistics" on page 499
- "Shared TS queue server: storage statistics" on page 500.

Shared TS queue server: coupling facility statistics

For queues that do not exceed 32K bytes, the data is included in the queue index; otherwise, it is stored as a separate list. The individual fields have the following meanings.

The statistics are described in detail in the DFHXQS1D data area.

Table 141. Shared TS queue server: coupling facility statistics

Statistic name	Field	Description
Structure		
	S1PREF	First part of structure name
	S1POOL	Poolname part of structure name
	S1CNPREF	Prefix for connection name
	S1CNSYSN	Own MVS system name from CVTSNAME
Size	S1SIZE	Current allocated size of the list structure.
Elem size	S1ELEMLN	Data element size, fullword, used for the structure.
Max size	S1SIZEMX	Maximum size to which this structure could be altered.
Lists		
Total	S1HDRS	Maximum number of list headers
Control	S1HDRSCT	Headers used for control lists
Data	S1HDRSQD	Headers available for queue data
In use	S1USEDCT	Number of entries on used list
Max used	S1USEDHI	Highest number of entries on used list
Entries		
In Use	S1ENTRCT	Number of entries currently in use.
Max Used	S1ENTRHI	Maximum number in use (since last reset).
Min Free	S1ENTRLO	Minimum number of free entries (since last reset).
Total	S1ENTRMX	Total data entries in the currently allocated structure. (Obtained at connection time, may be updated by ALTER).
	S1FREECT	Number of entries on free list
	S1ENTRRT	Entry size of entry to element ratio
	S1FREEHI	Highest number of entries on free list
Elements		
In use	S1ELEMCT	Number of elements currently in use.
Max used	S1ELEMHI	Maximum number in use (since last reset).
Min free	S1ELEMLO	Number of elements currently free (total minus used).

Table 141. Shared TS queue server: coupling facility statistics (continued)

Statistic name	Field	Description
Total	S1ELEMMX	Total data elements in the currently allocated structure. (Obtained at connection time, may be updated by ALTER).
	S1ELEMPW	Data element size, power of 2, used for the structure.
	S1ELEMPE	Maximum number of elements per entry (for 32K)
	S1ELEMRT	Element size of entry to element ratio.
Queues		
Current	S1INDXCT	Number of queues currently in existence.
Highest	S1INDXHI	Highest number of queues at any time (since last reset).
Index access counts		
Wrt adjs	S1WRACT	Number of index writes to update adjunct area only. (This area contains the read cursor for small queues and the queue status including last used data).
Inquires	S1INQCT	Inquire on queue index entry
Reads	S1RDQCT	Read queue index entry
Writes	S1WRQCT	Write queue index entry.
Deletes	S1DLQCT	Delete queue index entry.
Rereads	S1RRQCT	Number of index data reads which had to be repeated because the data was larger than the default data transfer size.
Data access counts		
Creates	S1CRLCT	Number of times a separate data list was created.
Writes	S1WRLCT	Number of queue writes (new or update) including data.
Reads	S1RDLCT	Number of queue index reads.
Deletes	S1DLLCT	Delete list (1 per overall delete).
Rereads	S1RRLCT	Number of list data reads which had to be repeated because the data was larger than the default data transfer size.
Rewrites	S1RWLCT	Rewrite list entry.
	S1INLCT	Inquire on list entry
Response Counts		
Asynch	S1ASYCT	Number of asynchronous requests.
Normal	S1RSP1CT	Number of normal responses.
Timeout	S1RSP2CT	Request timed out by the CF and should be restarted.
Not fnd	S1RSP3CT	Specified entry (queue or item) was not found.
Vers chk	S1RSP4CT	A version check failed for an entry being updated, indicating another task had updated it first.
List chk	S1RSP5CT	A list authority comparison failed, usually indicating big queue was deleted.
List full	S1RSP6CT	Maximum list key reached, indicating max queue size or max queues reached depending on list.
Str full	S1RSP7CT	The list structure is out of space.
I/O err	S1RSP8CT	An IXLLIST return code occurred other than those described above.

Shared TS queue server: buffer pool statistics

These statistics are for the queue index buffer pool, which is used to read and write queue index entries plus the associated data if the total queue size does not exceed 32K bytes. Buffers containing recently accessed queue index entries are added to a least recently used chain. This means that if another request for the same queue arrives shortly afterwards, it may be possible to optimize the processing based on the assumption that the copy in the buffer is probably already correct. If all other buffers are in use, a request for a new buffer will discard the contents of the least recently used buffer and reuse the storage as a free buffer. The queue server does not use some of the AXM management functions (such as KEEP or PURGE) so those counters will be zero. These fields describe the current state of the buffer pool.

The statistics are described in detail in the DFHXQS2D data area. The individual fields have the following meanings:

Table 142. Shared TS queue server:queue pool statistics

Statistic name	Field	Description
Buffers		
Total	S2BFQTY	Number of buffers in the pool.
Max used	S2BFENTH	Highest number ever used (not affected by reset).
Active	S2BFACTS	Buffers currently in use.
On LRU	S2BFLRUS	Buffers with valid contents on LRU chain to allow reuse.
Empty	S2BFEMPS	Buffers previously used but now empty.
Requests		
Gets	S2BFGETS	Requests to get a buffer.
Puts	S2BFPUTS	Put back buffer with valid contents
Keep	S2BFKEPS	Keeps (put back buffer with modified contents).
Free	S2BFFRES	Requests to put back a buffer as empty.
Purges	S2BFPURS	Request to discard contents of a previously valid buffer.
Results (Get)		
Got hit	S2BFHITS	Requests to put back a buffer with valid contents.
Got free	S2BFGFRS	Requests to put back a buffer with modified contents. (This function is not currently used by the queue server).
Got new	S2BFGNWS	Request obtained a buffer not previously used
Got LRU	S2BFGLRS	Request discarded and reused the oldest valid buffer.
No buf	S2BFGNBS	GETs which returned no buffer.
Errors		
Not freed	S2BFFNOS	A request tried to release a buffer it did not own. (This can occur during error recovery).
No purge	S2BFPNFS	A purge request did not find a matching buffer.
Not owned	S2BFPNOS	A purge request hit a buffer owned by another task.
Waits		
Pool lock	S2BFPWTS	Waits on buffer pool lock.
Buf lock	S2BFLWTS	GET wait on buffer lock.

Shared TS queue server: storage statistics

Storage in the AXMPGANY and AXMPGLOW pools is allocated in multiples of 4K pages on a 4K boundary. The most frequent use is for segments of LIFO stack storage. Storage is initially allocated from the pool using a bit map. For faster allocation, free areas are not normally returned to the pool but are added to a vector of free chains depending on the size of the free area (1 to 32 pages). When storage is being acquired, this vector is checked before going to the pool bit map. If there are no free areas of the right size and there is not enough storage left in the pool, free areas in the vector are put back into the pool, starting from the smallest end, until a large enough area has been created. This action appears as a compress attempt in the statistics. If there is still insufficient storage to satisfy the request, the request fails.

These statistics are for the named storage page pool produced since the most recent statistics (if any). Each of the storage statistics is shown in kilobytes and as a percentage of the total size.

The statistics are described in detail in the DFHXQS3D data area.

Table 143. Temporary storage data sharing: usage statistics

Statistic name	Field	Description
LOC=ANY storage pool statistics.		
Name	S3ANYNAM	Name of the storage pool AXMPGANY
Size	S3ANYSIZ	The total size of the storage pool.
	S3ANYPTR	Address of storage pool area
	S3ANYMX	Total pages in the storage pool
In Use	S3ANYUS	The amount of storage currently in use.
Free	S3ANYFR	The amount of storage within the pool that is currently free.
Min Free	S3ANYLO	The lowest amount of storage that has been free (since reset).
Gets	S3ANYRQG	The number of storage GET requests.
Fails	S3ANYRQF	The number of times that a storage request was unable to obtain the requested amount of storage even after a retry.
Frees	S3ANYRQS	The number of requests to release storage within the pool.
Retries	S3ANYRQC	The number of times that a storage request initially failed and was retried after merging any adjacent small free areas to form larger areas.
LOC=BELOW storage pool statistics.		
Name	S3LOWNAM	Name of the storage pool AXMPGLOW.
Size	S3LOWSIZ	The total size of the storage pool.
	S3LOWPTR	Address of the storage pool area.
	S3LOWMX	Total pages in the storage pool.
In Use	S3LOWUS	Number of used pages in the storage pool
Free	S3LOWFR	The amount of storage within the pool that is currently free.
Min Free	S3LOWLO	The lowest amount of storage that has been free.
Gets	S3LOWRQG	The number of requests to obtain storage within the pool.

Table 143. Temporary storage data sharing: usage statistics (continued)

Statistic name	Field	Description
Fails	S3LOWRQF	The number of times that a storage request was unable to obtain the requested amount of storage even after a retry.
Frees	S3LOWRQS	The number of requests to release storage within the pool.
Retries	S3LOWRQC	The number of times that a storage request initially failed and was retried after merging any adjacent small free areas to form larger areas.

Appendix C. Coupling facility data tables server statistics

This appendix provides information on statistics that are obtained for the coupling facility data tables server

This appendix describes the following:

- "Coupling facility data tables: list structure statistics"
- "Coupling facility data tables: table accesses statistics" on page 505
- "Coupling facility data tables: request statistics" on page 506
- "Coupling facility data tables: storage statistics" on page 507

Coupling facility data tables: list structure statistics

The statistics are described in detail in the DFHCFS6K data area.

The individual fields have the following meanings.

Table 144. Coupling facility data tables: list structure statistics

Statistic name	Field	Description
Structure		
	S6NAME	Full name of list structure
	S6PREF	First part of structure name
	S6POOL	Pool name part of structure name
	S6CNNAME	Name of connection to structure
	S6CNPREF	Prefix for connection name
	S6CNSYSN	Own MVS system name from CVTSNAME
Size	S6SIZE	Current allocated size of the list structure.
Max size	S6SIZEMX	Maximum size to which this structure could be altered.
Lists		
Total	S6HDRS	Maximum number of list headers in the structure.
Control	S6HDRSCT	Number of lists in use for control information.
Data	S6HDRSTD	Number of lists in use for table data.
Structure		
Elem size	S6ELEMLN	Data element size used for the structure.
	S6ELEMPW	Data element size as a power of 2
	S6ELEMRT	Element side of entry:element ratio
	S6ENTRRT	Entry side of entry:element ratio
Entries		
In use	S6ENTRCT	Number of entries currently in use.
Max used	S6ENTRHI	Maximum number in use (since last reset).
Min free	S6ENTRLO	Minumum number of free entries (since last reset).
Total	S6ENTRMX	Total entries in the currently allocated structure (initially set at structure connection time and updated on completion of any structure alter request).
Elements		
In Use	S6ELEMCT	Number of elements currently in use.
Max Used	S6ELEMHI	Maximum number in use (since last reset).

Table 144. Coupling facility data tables: list structure statistics (continued)

Statistic name	Field	Description
Min Free	S6ELEMLO	Minimum number of free elements (since last reset)
Total	S6ELEMMX	Total data elements in the currently allocated structure (initially set at structure connection time and updated on completion of any structure alter request).
List entry counts		
	S6USEVEC	Usage vector, five pairs of words
	S6USEDCT	Number of entries on used list
	S6USEDHI	Highest number of entries on used list
	S6FREECT	Number of entries on free list
	S6FREEHI	Highest number of entries on free list
	S6INDXCT	Number of entries in table index
	S6INDXHI	Highest entries in table index
	S6APPLCT	Number of entries in APPLID list
	S6APPLHI	Highest entries in APPLID list
	S6UOWLCT	Number of entries in UOW list
	S6UOWLHI	Highest entries in UOW list
Main type of CF request		
Table index lists		
Reads	S6RDICT	Number of table index reads.
Write	S6WRICT	Number of table index writes to create new tables.
Rewrite	S6RWICT	Number of table index writes to update table status.
Delete	S6DLICT	Number of table index deletes.
Data list controls		
Writes	S6CRLCT	Number of times a new data list was allocated.
Rewrites	S6MDLCT	Number of times data list controls were modified.
Deletes	S6DLLCT	Number of times a data list was deleted for reuse.
Table data record		
Reads	S6RDDCT	Number of data entry reads.
Writes	S6WRDCT	Number of data entry writes.
Rewrites	S6RWDCT	Number of data entry rewrites.
Deletes	S6DLDCT	Number of data entry deletes.
Data list controls		
Reads	S6INLCT	Inquire on data list
Lock release messages		
Reads	S6RDMCT	Number of lock release messages read by this server.
Writes	S6WRMCT	Number of lock release messages sent by this server.
UOW index list		
Reads	S6RDUCT	Number of UOW list reads.
Writes	S6WRUCT	Number of UOW list writes (usually at PREPARE)
Rewrites	S6RWUCT	Number of UOW list rewites (usually at COMMIT).
Deletes	S6DLUCT	Number of UOW list deletes (usually after COMMIT).
APPLID index lists		

Table 144. Coupling facility data tables: list structure statistics (continued)

Statistic name	Field	Description
Read	S6RDACT	Read APPLID entry
Write	S6WRACT	Write APPLID entry
Rewrite	S6RWACT	Rewrite APPLID entry
Delete	S6DLACT	Delete APPLID entry
Internal CF requests		
	S6RRLCT	Reread entry for full data length
Asynch	S6ASYCT	Number of requests for which completion was asynchronous.
IXLLIST completion		
Normal	S6RSP1CT	Number of normal responses.
Len err	S6RSP2CT	Entry data was larger than the inputbuffer length, which normally results in a retry with a larger buffer.
Not fnd	S6RSP3CT	The specified entry (table or item) was not found.
Vers chk	S6RSP4CT	A version check failed for an entry being updated, indicating that another task had updated it first.
List chk	S6RSP5CT	A list authority comparison failed, mismatch caused by table status update
List full	S6RSP6CT	A table reached the maximum number of items causing the relevant list to be marked as full.
Str full	S6RSP7CT	The list structure became full.
I/O err	S6RSP8CT	Some other error code was returned by IXLLIST.

Coupling facility data tables: table accesses statistics

These statistics are described in detail in the DFHCFS7K data area. The individual fields have the following meanings:

Table 145. Coupling facility data tables:queue pool statistics

Statistic name	Field	Description
Access		
	S7TABLE	Table name padded with spaces
Vector		
	S7STATS	Statistics vector
Table requests		
Open	S7OCOPEN	Number of successful OPEN requests for the table.
Close	S7OCCLOS	Number of successful CLOSE requests for the table.
Set Attr	S7OCSET	Number of times new table status was set.
Delete	S7OCDELE	Number of times the table of that name was deleted.
Stats	S7OCSTAT	Extract table statistics.
Record requests		
Point	S7RQPOIN	Number of POINT requests.
Highest	S7RQHIGH	Number of requests for current highest key.
Read	S7RQREAD	Number of READ requests (including those for UPDATE)

Table 145. Coupling facility data tables:queue pool statistics (continued)

Statistic name	Field	Description
Read del	S7RQRDDL	Number of combined READ and DELETE requests.
Unlock	S7RQUNLK	Number of UNLOCK requests.
Loads	S7RQLOAD	Number of records written by initial load requests.
Write	S7RQWRIT	Number of WRITE requests for new records.
Rewrite	S7RQREWR	Number of REWRITE requests.
Delete	S7RQDELE	Number of DELETE requests
Del Mult	S7RQDELM	Number of multiple (generic) delete requests.

Coupling facility data tables: request statistics

These statistics are described in detail in the DFHCFS8K data area. The individual fields have the following meanings:

Table 146. Coupling facility data tables:request statistics

Statistic name	Field	Description
Vector		
	S8STATS	Statistics vector
Table		
Open	S8OCOPEN	Number of successful OPEN requests for the table
Close	S8OCCLOS	Number of successful CLOSE requests for the table.
Set Attr	S8OCSET	Number of times new table status was set.
Delete	S8OCDELE	Number of times the table of that name was deleted.
Stats	S8OCSTAT	Number of times table access statistics were extracted.
Record		
Point	S8RQPOIN	Number of POINT requests.
Highest	S8RQHIGH	Number of requests for current highest key
Read	S8RQREAD	Number of READ requests (including those for UPDATE)
Read Del	S8RQRDDL	Number of combined READ and DELETE requests
Unlock	S8RQUNLK	Number of UNLOCK requests.
Loads	S8RQLOAD	Number of records written by initial load requests.
Write	S8RQWRIT	Number of WRITE requests for new records
Rewrite	S8RQREWR	Number of REQRITE requests.
Delete	S8RQDELE	Number of DELETE requests.
Del Mult	S8RQDELM	Number of multiple (generic) delete requests
Table		
Inquire	S8IQINQU	Number of INQUIRE table requests.
UOW		
Prepare	S8SPPREP	Number of units of work prepared.
Retain	S8SPRETA	Number of units of work whose locks were retained.
Commit	S8SPCOMM	Number of units of work committed.
Backout	S8SPBACK	Number of units of work backed out.
Inquire	S8SPINQU	Number of units of work INQUIRE requests.

Table 146. Coupling facility data tables:request statistics (continued)

Statistic name	Field	Description
Restart	S8SPREST	Number of times recoverable connections were restarted.

Coupling facility data tables: storage statistics

These statistics are returned by the AXM page pool management routines for the pools AXMPGANY and AXMPGLOW. Storage in these pools is allocated in multiples of 4K pages on a 4K boundary. The most frequent use is for segments of LIFO stack storage.

Storage is initially allocated from the pool using a bit map. For faster allocation, free areas are not normally returned to the pool but are added to a vector of free chains depending on the size of the free area (1 to 32 pages). When storage is being acquired, this vector is checked before going to the pool bit map.

If there are no free areas of the right size and there is not enough storage left in the pool, free areas in the vector are put back into the pool, starting from the smallest end, until a large enough area has been created. This action appears as a compress attempt in the statistics. If there is still insufficient storage to satisfy the request, the request fails.

The statistics are described in detail in the DFHCFS9K data area.

Table 147. Coupling facility data tables: storage statistics

Statistic name	Field	Description
LOC=ANY storage pool statistics.		
Name	S9ANYNAM	Name of the storage pool AXMPGANY
Size	S9ANYSIZ	Size of the storage pool area
	S9ANYPTR	Address of storage pool area
	S9ANYMX	Total pages in the storage pool
In Use	S9ANYUS	Number of used pages in the pool
Free	S9ANYFR	Number of free pages in the pool
Min Free	S9ANYLO	Lowest free pages (since reset)
Gets	S9ANYRQG	Storage GET requests.
Fails	S9ANYRQF	GETs which failed to obtain storage
Frees	S9ANYRQS	Storage FREE requests
Retries	S9ANYRQC	Compress (defragmentation) attempts
LOC=BELOW storage pool statistics.		
Name	S9LOWNAM	Pool name AXMPGLOW
Size	S9LOWSIZ	Size of storage pool area
	S9LOWPTR	Address of storage pool area
	S9LOWMX	Total pages in the storage pool
In Use	S9LOWUS	Number of used pages in the storage pool
Free	S9LOWFR	Number of free pages in the storage pool
Min Free	S9LOWLO	Lowest free pages (since reset)
Gets	S9LOWRQG	Storage GET requests
Fails	S9LOWRQF	GETs which failed to obtain storage
Frees	S9LOWRQS	Storage FREE requests
	S9LOWRQC	Compress (defragmentation) attempts

Appendix D. Named counter sequence number server

This appendix provides information on statistics that are obtained for the named counter sequence number server.

This appendix describes the following:

- "Named counter sequence number server statistics"
- "Named counter server: storage statistics" on page 510

Named counter sequence number server statistics

The statistics are described in detail in the DFHNCS4K data area.

The individual fields have the following meanings.

Table 148. Named counter server: list structure statistics

Statistic name	Field	Description
Structure:		
Lists		
	S4NAME	Full name of list structure
	S4PREF	First part of structure name
	S4POOL	Pool name part of structure name
	S4CNNAME	Name for connection to structure
	S4CNPREF	Prefix for connection name
	S4CNSYSN	Own MVS system name from CVTSNAME
Size	S4SIZE	Current allocated size for the list structure.
Max size	S4SIZEMX	Maximum size to which this structure could be altered.
Entries		
In Use	S4ENTRCT	Number of entries currently in use.
Max Used	S4ENTRHI	Maximum number of entries in use (since last reset).
Min Free	S4ENTRLO	Minimum number of free entries (since last reset).
Total	S4ENTRMX	Total entries in the currently allocated structure (initially set at structure connection time and updated on completion of any structure alter request).
Requests		
Create	S4CRECT	Create counter
Get	S4GETCT	Get and increment counter
Set	S4SETCT	Set counter
Delete	S4DELCT	Delete counter
Inquire	S4KEQCT	Inquire KEQ
Browse	S4KGECT	Inquire KGE
Responses		
Asynch	S4ASYCT	Number of requests for which completion was asynchronous.
Normal	S4RSP1CT	Number of normal responses.
Not Fnd	S4RSP2CT	The specified entry (table or item) was not found.

Table 148. Named counter server: list structure statistics (continued)

Statistic name	Field	Description
Vers Chk	S4RSP3CT	A version check failed for an entry being updated, indicating that another task had updated it first.
List Chk	S4RSP4CT	A list authority comparison failed, usually meaning that the table is in the process of being deleted.
Str Full	S4RSP5CT	The list structure became full.
I/O Err	S4RSP6CT	Some other error code was returned by IXLLIST.

Named counter server: storage statistics

These are statistics returned by the AXM page pool management routines for the pools AXMPGANY and AXMPGLOW. Storage in these pools is allocated in multiples of 4K pages on a 4K boundary. The most frequent use is for segments of LIFO stack storage.

Storage is initially allocated from the pool using a bit map. For faster allocation, free areas are not normally returned to the pool but are added to a vector of free chains depending on the size of the free area (1 to 32 pages). When storage is being acquired, this vector is checked before going to the pool bit map.

If there are no free areas of the right size and there is not enough storage left in the pool, free areas in the vector are put back into the pool, starting from the smallest end, until a large enough area has been created. This action appears as a compress attempt in the statistics. If there is still insufficient storage to satisfy the request, the request fails.

These statistics are for the named storage page pool produced since the most recent statistics (if any). Each of the storage statistics is shown in kilobytes and as a percentage of the total size.

The statistics are described in detail in the DFHNCS5K data area.

Table 149. Temporary storage data sharing: usage statistics

Statistic name	Field	Description
LOC=ANY storage pool statistics		
Name	S5ANYNAM	Pool name AXMPGANY
Size	S5ANYSIZ	Size of the storage pool area
	S5ANYPTR	Address of storage pool area
	S5ANYMX	Total pages in the storage pool
In Use	S5ANYUS	Number of used pages in the pool
Free	S5ANYFR	Number of free pages in the pool
Min Free	S5ANYLO	The lowest free pages (since reset)
Gets	S5ANYRQG	Storage GET requests
Fails	S5ANYRQF	GETs which failed to obtain storage
Frees	S5ANYRQS	Storage FREE requests
Retries	S5ANYRQC	Compress (defragmentation) attempts
LOC=BELOW storage pool statistics		
Name	S5LOWNAM	Pool name AXMPGLOW
Size	S5LOWSIZ	Size of the storage pool area
	S5LOWPTR	Address of the storage pool area

Table 149. Temporary storage data sharing: usage statistics (continued)

Statistic name	Field	Description
	S5LOWMX	Total pages in the storage pool
In Use	S5LOWUS	Number of used pages in the storage pool
Free	S5LOWFR	Number of free pages in the storage pool
Min Free	S5LOWLO	The lowest number of free pages (since reset)
Gets	S5LOWRQG	Storage GET requests
Fails	S5LOWRQF	GETs which failed to obtain storage
Frees	S5LOWRQS	Storage FREE requests
Retries	S5LOWRQC	Compress (defragmentation) attempts

Appendix E. The sample statistics program, DFH0STAT

Use the statistics sample program, DFH0STAT, to help you determine and adjust the values needed for CICS storage parameters; for example, using DSALIMIT and EDSALIMIT. The program produces a report showing critical system parameters from the CICS dispatcher, an analysis of the CICS storage manager and loader statistics, and an overview of the MVS storage in use. The program demonstrates the use of the EXEC CICS INQUIRE and EXEC CICS COLLECT STATISTICS commands to produce an analysis of a CICS system. You can use the sample program as provided or modify it to suit your needs.

The sample statistics program consists of the following resources:

DFH0STAT
Statistics sample program, VS COBOL II release 2 or later.

DFH$STAS
Assembler language program called by DFH0STAT.

DFH$STCN
Assembler language program called by DFH0STAT.

DFH$STTB
Assembler language program loaded by DFH0STAT.

DFH0STM
Mapset used by the STAT transaction.

DFH0STS
Mapset used by the STAT transaction.

STAT
Transaction used to invoke the program, DFH0STAT.

All programs are command level and run above the 16MB line.

DFH0STAT can be invoked from the PLT at PLTPI (second phase) or PLTSD (first phase) or as a CICS transaction (either from a console or as a conversational transaction from a terminal). The output is sent via the CICS JES SPOOL interface for which a number of default parameters can be changed by the user to specify the distribution of the report(s). These defaults are defined in the working-storage section of this program under the 01 level “OUTPUT-DEFAULTS”.

If an EXEC CICS SPOOL .. command fails when the program is run as a transaction, an error message is displayed on the users screen and the transaction will continue. If the program is not being run from a terminal, a message will be sent to the console using EXEC CICS WRITE OPERATOR commands and the transaction will be terminated normally.

To enable you to use the sample program, you must:.

1. Assemble and link-edit the BMS mapsets DFH0STM and DFH0STS. Include the physical mapset in a library that is in the DFHRPL concatenation. You can either include the symbolic mapset in a user copy library or insert it directly into the source of DFH0STAT.

2. Translate the DFH0STAT program source code, turning CICS commands into code understood by the compiler. The program source code is provided in CICSTS13.CICS.SDFHSAMP.
3. Compile the translator output for DFH0STAT to produce object code.
4. Link-edit the object module to produce a load module, which you store in an application load library that is concatenated to the DFHRPL DD statement of the CICS startup job stream.
5. Create resource definition entries, in the CSD, for the programs, mapset, and the STAT transaction. The CICS-supplied RDO group, DFH$STAT, contains all the necessary resource definitions for this sample.
6. Define the system initialization parameter SPOOL=YES. This specifies that you need support for the system spooling interface.

Analyzing DFH0STAT Reports

The sample statistics program DFH0STAT can produce reports about:

- System status, monitoring, statistics, trace and dump
- Transaction manager
- Dispatcher
- Dispatcher TCBs
- Storage below and above 16MB
- Loader
- Selected storage subpools
- Transaction classes
- Transactions
- Transaction totals including subspace usage information
- Programs
- Program totals
- DFHRPL analysis
- Programs by DSA and LPA
- Temporary storage
- Temporary storage queues
- Tsqueue totals
- Temporary storage queues by shared ts pool
- Transient data
- Transient data queues
- Tdqueue totals
- Journalnames
- Logstreams
- Program autoinstall, terminal autoinstall, and VTAM
- Connections and modenames
- TCP/IP services
- LSR pools
- Files
- Data tables
- Coupling facility data table pools
- Exit programs and global user exits

- DB2 connection and DB2 entries
- Enqueue manager
- Recovery manager
- Page index.

In addition, the statistics report selection mapset allows the user to select the required statistics reports. Figure 34 shows an example of the statistics report selection mapset with the default reports selected.

```
          Sample Program - CICS Statistics Print Report Selection
                                                    08/24/1998  12:02:55

Select the statistics reports required and press Enter

 System Status. . . . . . . . . . . Y   Page Index . . . . . . . . . . . . Y
 Transaction Manager/Dispatcher . . Y
 Storage Manager. . . . . . . . . . Y   Loader and Storage Subpools. . . . Y
 Transactions . . . . . . . . . . . Y   Transaction Classes. . . . . . . . N
 Programs and DFHRPL Analysis . . . Y   Programs by DSA and LPA. . . . . . N
 Temporary Storage. . . . . . . . . Y   Temporary Storage Queues . . . . . N
                                        Temporary Storage Queues by Pool . N
 Transient Data . . . . . . . . . . Y   Transient Data Queues. . . . . . . N
 Journals and Logstreams. . . . . . Y
 Autoinstall and VTAM . . . . . . . Y   Connections and Modenames. . . . . N
                                        TCP/IP Services. . . . . . . . . . N
 Files. . . . . . . . . . . . . . . N
 LSR pools. . . . . . . . . . . . . N   CF Data Table pools. . . . . . . . N
 DB2 Connection and Entries . . . . N   User Exit Pgms/Global User Exits . N
 Enqueue Manager. . . . . . . . . . N   Recovery Manager . . . . . . . . . N



 F3=Return to print
```

Figure 34. statistics report selection screen

The heading of each report includes the generic applid, sysid, jobname, date and time, and the CICS version and release information.

System Status Report

Figure 35 on page 516 shows the format of the System status report. The field headings and contents are described in Table 150 on page 516.

```
Applid IYK2Z1V3  Sysid CJB3  Jobname CI07CJB3    Date 12/04/1998  Time 08:34:31      CICS 5.3.0     PAGE     1

System Status

  MVS Product Name. . . . :  MVS/SP6.0.5

  CICS Startup. . . . . . :  COLD
  CICS Status . . . . . . :  ACTIVE                              RLS Status. . . . . . . . . . . . :  RLS=NO

  Storage Protection. . . :  ACTIVE                              RRMS/MVS Status . . . . . . . . . :  OPEN
  Transaction Isolation . :  ACTIVE
  Reentrant Programs. . . :  PROTECT                             VTAM Open Status. . . . . . . . . :  OPEN
                                                                 IRC Status. . . . . . . . . . . . :  OPEN
  Program Autoinstall . . . . . :  INACTIVE                      TCP/IP Status . . . . . . . . . . :  OPEN
  Terminal Autoinstall. . . . . :  ENABLED
                                                                 WEB Garbage Collection Interval . :      60
  Activity Keypoint Frequency. . . . . . . :       4,000       Terminal Input Timeout Interval . :       5
  Logstream Deferred Force Interval. . . . :          30

  DB2 Connection Name . . . . . :  DB3A
  DB2 Connection Status . . . . :  NOTCONNECTED


Monitoring                                      Statistics

  Monitoring . . . . . :  ON                                   Statistics Recording . . . . . :  ON
  Exception Class. . . :  ON                                   Statistics Last Reset Time . . :  08:20:17
  Performance Class. . :  ON                                   Elapsed Time Since Reset . . . :  00:14:13

  Exception Class Records . . . . . :          0               Statistics Interval. . . . . . :  01:00:00
  Exception Records Suppressed. . . :          0               Next Statistics Collection . . :  09:00:00
  Performance Class Records . . . . :         44               Statistics End-of-Day Time . . :  00:00:00
  Performance Records Suppressed. . :          0
  SMF Records . . . . . . . . . . . :          2
  SMF Errors. . . . . . . . . . . . :          0


Trace Status                                    Dumps

  Internal Trace Status . . . . :  STARTED                     System Dumps . . . . . . . . . . :           0
  Auxiliary Trace Status. . . . :  STOPPED                     System Dumps Suppressed. . . . . :           0
  GTF Trace Status. . . . . . . :  STOPPED
                                                               Transaction Dumps. . . . . . . . :           0
  Internal Trace Table Size . . :    2,000K                    Transaction Dumps Suppressed . . :           0
  Current Auxiliary Dataset . . :  A
  Auxiliary Switch Status . . . :  NOSWITCH
```

Figure 35. The System Status Report

Table 150. Fields in the System Status Report

Field Heading	Description
System Status	
MVS Product Name	Indicates the product level of MVS. Source field: MVS field CVTPRODN
CICS Startup	Indicates the type of CICS startup. Source field: EXEC CICS INQUIRE SYSTEM STARTUP(cvda) COLDSTATUS(cvda)

Table 150. Fields in the System Status Report (continued)

Field Heading	Description
CICS Status	Indicates the current status of the local CICS system. Source field: EXEC CICS INQUIRE SYSTEM CICSSTATUS(cvda)
RLS Status	Indicates the current status of VSAM RLS for this CICS system. Source field: EXEC CICS INQUIRE SYSTEM RLSSTATUS(cvda)
Storage Protection	Indicates the status of storage protection. Source field: EXEC CICS INQUIRE SYSTEM STOREPROTECT(cvda)
RRMS/MVS Status	Indicates the current status of RRMS/MVS for this CICS system. Source field: EXEC CICS INQUIRE RRMS OPENSTATUS(cvda)
Transaction Isolation	Indicates the status of transaction isolation. Source field: SMSTRANISO
Reentrant Programs	Indicates if read-only programs reside in key-0 protected storage. Source field: SMSRENTPGM
VTAM Open Status	Indicates the current status of the VTAM connection for this CICS system. Source field: EXEC CICS INQUIRE VTAM OPENSTATUS(cvda)
IRC Status	Indicates the current status of IRC for this CICS system. Source field: EXEC CICS INQUIRE IRC OPENSTATUS(cvda)
Program Autoinstall	Indicates the current status of program autoinstall. Source field: EXEC CICS INQUIRE SYSTEM PROGAUTOINST(cvda)
TCP/IP Status	Indicates the current status of TCP/IP for this CICS system. Source field: EXEC CICS INQUIRE TCPIP OPENSTATUS(cvda)
Terminal Autoinstall	Indicates the current status of terminal autoinstall. Source field: EXEC CICS INQUIRE AUTOINSTALL(cvda)
Activity Keypoint Frequency	The current activity keypoint trigger value, which is the number of logging operations between the taking of keypoints. Source field: EXEC CICS INQUIRE SYSTEM AKP(data area).
Garbage Collection Interval	Is the current interval at which the Web garbage collection task runs to clean up Web 3270 state data. Source field: EXEC CICS INQUIRE WEB GARBAGEINT()
Logstream Deferred Force Interval	Is the current logstream deferred force interval Source field: EXEC CICS INQUIRE SYSTEM LOGDEFER()
Terminal Input Timeout Interval	Is the current period of time after which inactive Web 3270 sessions are eligible for garbage collection. Source field: EXEC CICS INQUIRE WEB TIMEOUTINT()
DB2 Connection Name	The name of the currently installed DB2 connection. Source field: EXEC CICS INQUIRE SYSTEM DB2CONN(data area)

Table 150. Fields in the System Status Report (continued)

Field Heading	Description
DB2 Connection Status	The current status of the CICS-DB2 Connection. Source field: EXEC CICS INQUIRE DB2CONN() CONNECTST(cvda)
Monitoring	
Monitoring	Indicates whether CICS monitoring is active in the system. Source field: EXEC CICS INQUIRE MONITOR STATUS(cvda)
Exception Class	Indicates whether the exception class of CICS monitoring data is being collected. Source field: EXEC CICS INQUIRE MONITOR EXCEPTCLASS(cvda)
Performance Class	Indicates whether the performance class of CICS monitoring data is being collected. Source field: EXEC CICS INQUIRE MONITOR PERFCLASS(cvda)
Subsystem ID	The 4-character name used as the subsystem identification in the monitoring sysevent class records. Source field: EXEC CICS INQUIRE MONITOR SUBSYSTEMID(4-character data-area)
Exception Class Records	The number of exception records written to SMF. Source field: MNGER
Exception Class Suppressed	The number of exception records suppressed by a global user exit program at exit point XMNOUT. Source field: MNGERS
Performance Class Records	The number of performance records scheduled for output to SMF. Because the monitoring domain buffers performance class records, there is still a possibility that monitoring can be deactivated, thus preventing the output of the performance class records that have been buffered. Source field: MNGPR
Performance Class Suppressed	The number of performance records suppressed by a global user exit program at exit point XMNOUT. Source field: MNGPRS
SMF Records	The number of SMF records written to the SMF data set. CICS writes exception class SMF records as soon as the monitor domain is notified of the exception completion, so there is one exception record per SMF record. The performance class, however, has many performance class records per SMF record. The SMF record for the performance class is written when the buffer is full, performance class has been deactivated, or CICS is quiescing. Source field: MNGSMFR
SMF Errors	The number of non-OK responses from the request to write a record to SMF. This count is incremented when an SMF write fails for any reason, for example, when SMF is inactive. Source field: MNGSMFE
Statistics	

Table 150. Fields in the System Status Report (continued)

Field Heading	Description
Statistics Recording	The current status of statistics recording. Source field: EXEC CICS INQUIRE STATISTICS RECORDING(cvda)
Statistics Last Reset Time	The time of the last statistics reset. Source field: EXEC CICS COLLECT STATISTICS LASTRESET()
Elapsed Time Since Reset	The elapsed time since the last statistics reset. Source field:
Statistics Interval	The current statistics recording interval. Source field: EXEC CICS INQUIRE STATISTICS INTERVAL
Next Statistics Collection	The next statistics recording time. Source field: EXEC CICS INQUIRE STATISTICS NEXTTIME
Statistics End-of-Day Time	The current end-of-day time for recording statistics. Source field: EXEC CICS INQUIRE STATISTICS ENDOFDAY
Trace Status	
Internal Trace Status	The current status of internal tracing. Source field: EXEC CICS INQUIRE TRACEDEST INTSTATUS(cvda)
Auxiliary Trace Status	The current status of auxiliary tracing. Source field: EXEC CICS INQUIRE TRACEDEST AUXSTATUS(cvda)
GTF Trace Status	The current status of gtf tracing. Source field: EXEC CICS INQUIRE TRACEDEST GTFSTATUS(cvda)
Internal Trace Table Size	The current size of the internal trace table. Source field: EXEC CICS INQUIRE TRACEDEST TABLESIZE
Current Auxiliary Dataset	The name of the current auxiliary trace dataset. Source field: EXEC CICS INQUIRE TRACEDEST CURAUXDS(cvda)
Auxiliary Switch Status	The current status of the auxiliary trace autoswitch facility. Source field: EXEC CICS INQUIRE TRACEDEST SWITCHSTATUS(cvda)
Dumps	
System Dumps	The number of system dumps taken. Source field: SDGSDREQ
System Dumps Suppressed	The number of system dumps suppressed. Source field: SDGSDSUP
Transaction Dumps	The number of transaction dumps taken. Source field: SDGTDREQ
Transaction Dumps Suppressed	The number of transaction dumps suppressed. Source field: SDGTDSUP

Transaction Manager Report

Figure 36 shows the format of the Transaction manager report. This report is produced using the EXEC CICS COLLECT STATISTICS TRANSACTION command. The statistics data is mapped by the DFHXMGDS DSECT. The field headings and contents are described in Table 151.

```
Applid IYK2Z1V3  Sysid CJB3  Jobname CI07CJB3    Date 08/20/1998  Time 15:33:57     CICS 5.3.0    PAGE  2


Transaction Manager

  Total Accumulated transactions so far. . :              58

  Accumulated transactions (since reset) . :               9          Transaction Rate per second. . :     0.00

  Maximum transactions allowed (MXT) . . . :              75
  Times at MXT . . . . . . . . . . . . . . :               0
  Current Active User transactions . . . . :               1
  Peak Active User transactions. . . . . . :               1
  Total Active User transactions . . . . . :               9

  Current Running transactions . . . . . . :               1
  Current Dispatchable transactions. . . . :               0
  Current Suspended transactions . . . . . :               0
  Current System transactions. . . . . . . :               0

  Transactions Delayed by MXT. . . . . . . :               0
  Total MXT queueing time. . . . . . . . . :  00:00:00.00000
  Average MXT queueing time. . . . . . . . :  00:00:00.00000

  Current Queued User transactions . . . . :               0
  Total Queueing time for current queued . :  00:00:00.00000
  Average Queueing time for current queued :  00:00:00.00000
```

Figure 36. The Transaction Manager Report

Table 151. Fields in the Transaction Manager Report

Field Heading	Description
Transaction Manager	
Accumulated transactions so far	The total number of tasks that have accumulated so far. Source field: (XMGTNUM + XMGNUM)
Accumulated transactions (since reset)	The number of tasks that have accumulated. Source field: XMGNUM
Transaction Rate per second	The number of transactions per second. Source field: (XMGNUM / Elapsed seconds since reset)
Maximum transactions allowed (MXT)	The specified maximum number of user transactions as specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM MAXTASKS(value) or EXEC CICS SET SYSTEM MAXTASKS(fullword binary data-value) commands. Source field: XMGMXT

Table 151. Fields in the Transaction Manager Report (continued)

Field Heading	Description
Times at MXT	The number of times that the number of active user transactions equalled the specified maximum number of user transactions (MXT). Source field: XMGTAMXT
Current Active User transactions	The current number of active user transactions. Source field: XMGCAT
Peak Active User transactions	The peak number of active user transactions reached. Source field: XMGPAT
Total Active User transactions	The total number of user transactions that have become active. Source field: XMGTAT
Current Running transactions	The current number of Running transactions. Source field: EXEC CICS INQUIRE TASKLIST RUNNING
Current Dispatchable transactions	The current number of Dispatchable transactions. Source field: EXEC CICS INQUIRE TASKLIST DISPATCHABLE
Current Suspended transactions	The current number of Suspended transactions. Source field: EXEC CICS INQUIRE TASKLIST SUSPENDED
Current System transactions	The current number of system transactions. Source field: ((Running + Dispatchable + Suspended) - XMGCAT)
Transactions Delayed by MXT	The number of user transactions that had to queue for MXT reasons before becoming active, excluding those still waiting. Source field: XMGTDT
Total MXT Queueing Time	The total time spent waiting by those user transactions that had to wait for MXT reasons. **Note:** This does not include those transactions still waiting. Source field: XMGTQTME
Average MXT Queueing Time	The average time spent waiting by those user transactions that had to wait for MXT reasons. Source field: (XMGTQTME / XMGTDT)
Current Queued User transactions	The current number of user transactions currently queuing for MXT reasons. **Note:** That this does not include transactions currently queued for Transaction Class. Source field: XMGCQT
Total Queueing Time for current queued	The total time spent waiting by those user transactions currently queued for MXT reasons. **Note:** This does not include the time spent waiting by those transactions that have finished queuing. Source field: XMGCQTME
Average Queueing Time for current queued	The average time spent waiting by those user transactions currently queued for MXT reasons. Source field: (XMGCQTME / XMGCQT)

Dispatcher Report

Figure 37 shows the format of the Dispatcher report. This report is produced using a combination of the EXEC CICS INQUIRE SYSTEM and EXEC CICS COLLECT STATISTICS DISPATCHER commands. The statistics data is mapped by the DFHDSGDS DSECT. The field headings and contents are described in Table 152.

```
Applid IYK2Z1V3  Sysid CJB3  Jobname CI07CJB3    Date 08/20/1998  Time 15:33:57   CICS 5.3.0   PAGE    3


Dispatcher

  Current ICV time . . . . . :       5,000ms
  Current ICVR time. . . . . :       6,000ms
  Current ICVTSD time. . . . :         500ms
  Current PRTYAGING time . . :         500ms

  Force Quasi-Reentrant. . . :  No


  Current CICS Dispatcher tasks. . . . . :          15
  Peak CICS Dispatcher tasks . . . . . :            15

  Max Open TCB Limit (MAXOPENTCBS) . . . . :        15

  Current Open TCBs attached . . . . . :             0
  Peak Open TCBs attached. . . . . :                 0

  Current Open TCBs in use . . . . . :               0
  Peak Open TCBs in use. . . . . :                   0

  Times at Max Open TCB Limit. . . . . :             0

  Attaches Delayed by Max Open TCB Limit . :                0
  Total MAXOPENTCBS delay time . . . . . :  00:00:00.00000
  Average MAXOPENTCBS delay time . . . . . :  00:00:00.00000

  Current Attaches Delayed by MAXOPENTCBS. :                0
  Peak Attaches Delayed by MAXOPENTCBS . . :                0
  Total Delay time for current delayed . . :  00:00:00.00000
  Average Delay time for current delayed . :  00:00:00.00000
```

Figure 37. The Dispatcher Report

Table 152. Fields in the Dispatcher Report

Field Heading	Description
Dispatcher	
Current ICV time	The ICV time value (expressed in *milliseconds*) specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM TIME(value) or EXEC CICS SET SYSTEM TIME(fullword binary data-value) commands. Source field: DSGICVT
Current ICVR time	The current task runaway time interval. Source field: DSGICVRT

Table 152. Fields in the Dispatcher Report (continued)

Field Heading	Description
Current ICVTSD time	The ICVTSD time value (expressed in *milliseconds*) specified in the SIT, or as an override, or changed dynamically using CEMT SET SYSTEM SCANDELAY(value) or EXEC CICS SET SYSTEM SCANDELAY(fullword binary data-value) commands. Source field: DSGICVSD
Current PRTYAGING time	The current task priority aging factor. Source field: DSGPRIAG
Force Quasi-Reentrant	Indicates whether CICS will force all user application programs specified as CONCURRENCY(THREADSAFE) to run under the CICS QR TCB. Source field: EXEC CICS INQUIRE SYSTEM FORCEQR(cvda)
Current CICS Dispatcher tasks	The current number of tasks in the system. This figure includes all system tasks and all user tasks. Source field: DSGCNT
Peak CICS Dispatcher tasks	The peak number of tasks concurrently in the system. Source field: DSGPNT
Max Open TCB Limit (MAXOPENTCBS)	The current MAXOPENTCBS value (expressed as the number of open TCBs) as specified in the SIT, or as an override, or changed dynamically using the CEMT SET SYSTEM MAXOPENTCBS(value) or EXEC CICS SET SYSTEM MAXOPENTCBS(fullword binary data-value) commands. Source field: EXEC CICS INQUIRE SYSTEM MAXOPENTCBS()
Current Open TCBs attached	The current number of open TCBs attached. Source field: DSGCNUAT
Peak Open TCBs attached	The peak number of open TCBs attached. Source field: DSGPNUAT
Current Open TCBs in use	The current number of open TCBs in use. Source field: DSGCNUUS
Peak Open TCBs in use	The peak number of open TCBs in use. Source field: DSGPNUUS
Times at Max Open TCB Limit	The number of times the system reached the MAXOPENTCBS limit. Source field: DSGNTCBL
Attaches Delayed by Max Open TCB Limit	The total number of TCB attaches delayed because the system had reached the MAXOPENTCBS limit. Source field: DSGTOTNW
Total MAXOPENTCBS delay time	The total time that open mode TCBs were delayed because the system had reached the MAXOPENTCBS limit. Source field: DSGTOTWL

Table 152. Fields in the Dispatcher Report (continued)

Field Heading	Description
Average MAXOPENTCBS delay time	The average time that an open mode TCB attach was delayed because the system had reached the MAXOPENTCBS limit. Source field: (DSGTOTWL / DSGTOTNW)
Current Attaches Delayed by MAXOPENTCBS	The current number of TCB attaches that are delayed because the system is at the MAXOPENTCBS limit. Source field: DSGCURNW
Peak Attaches delayed by MAXOPENTCBS	The peak number of TCB attaches delayed because the system is at the MAXOPENTCBS limit. Source field: DSGPEANW
Total Delay Time for Current Delayed	The current total delay time for the TCB attaches that are delayed because the system is at the MAXOPENTCBS limit. Source field: DSGCURWT
Average Delay time for current delayed	The average delay time for the TCB attaches that are currently delayed because the system is at the MAXOPENTCBS limit. Source field: (DSGCURWT / DSGCURNW)

Dispatcher TCBs Report

Figure 38 on page 525 shows the format of the Dispatcher TCBs report. This report is produced using the EXEC CICS COLLECT STATISTICS DISPATCHER command. The statistics data is mapped by the DFHDSGDS DSECT. The field headings and contents are described in Table 153 on page 525.

```
Applid IYK2Z1V3  Sysid CJB3  Jobname CI07CJB3    Date 12/02/1998  Time 08:52:21                CICS 5.3.0                PAGE     4

Dispatcher TCBs

  Dispatcher Start Time and Date . . . . . :  08:00:23.71167  12/02/1998

  Address Space Accumulated CPU Time . . . :  00:00:06.60362     (Not Reset)
  Address Space Accumulated SRB Time . . . :  00:00:00.65995     (Not Reset)

  Address Space CPU Time (Since Reset) . . :  00:00:06.52665
  Address Space SRB Time (Since Reset) . . :  00:00:00.65097


  Number of CICS TCB MODEs . :          11


   TCB   Current    Peak      TCBs     <--------- TCBs Detached -------->
   Mode    TCBs     TCBs    Attached   | Unclean     Stolen       Other |  TCB Steals
  ____________________________________________________________________________________
    QR         1        1           1         N/A         N/A         N/A         N/A
    RO         1        1           1         N/A         N/A         N/A         N/A
    CO         1        1           1         N/A         N/A         N/A         N/A
    SZ         1        1           1         N/A         N/A           0         N/A
    RP         0        0           0         N/A         N/A           0         N/A
    FO         1        1           1         N/A         N/A         N/A         N/A
    SL         1        1           1         N/A         N/A           0         N/A
    SO         1        1           1         N/A         N/A           0         N/A
    J8         0        0           0           0           0           0           0
    L8         0        0           0           0           0           0           0
    S8         0        0           0           0           0           0           0
  ____________________________________________________________________________________
  Totals       7                    7           0           0           0           0


   TCB   Current    Peak   Op. System    Op. System      Total TCB       Total TCB         DS TCB      TCB CPU/Disp
   Mode    TCBs     TCBs     Waits        Wait Time     Dispatch Time     CPU Time         CPU Time         Ratio
  ____________________________________________________________________________________________________________

    QR         1        1       3,492  00:51:51.09861  00:00:06.61393  00:00:03.69497  00:00:00.47251      55.8%
    RO         1        1         123  00:51:44.37901  00:00:11.83684  00:00:01.46320  00:00:00.00512
    CO         1        1         484  00:29:28.25675  00:00:00.71757  00:00:00.28578  00:00:00.16471
    SZ         1        1         436  00:29:08.30509  00:00:01.37113  00:00:00.59021  00:00:00.21527
    RP         0        0           0  00:00:00.00000  00:00:00.00000  00:00:00.00000  00:00:00.00000
    FO         1        1          87  00:00:23.01983  00:00:03.76582  00:00:00.29225  00:00:00.00414
    SL         1        1           4  00:31:28.23371  00:00:00.00676  00:00:00.00593  00:00:00.00009
    SO         1        1           2  00:00:15.28525  00:00:01.20830  00:00:00.00000  00:00:00.00000
    J8         0        0           0  00:00:00.00000  00:00:00.00000  00:00:00.00000  00:00:00.00000
    L8         0        0           0  00:00:00.00000  00:00:00.00000  00:00:00.00000  00:00:00.00000
    S8         0        0           0  00:00:00.00000  00:00:00.00000  00:00:00.00000  00:00:00.00000
  ____________________________________________________________________________________________________________
  Totals                                                               00:00:06.33236  00:00:00.86185
```

Figure 38. The Dispatcher TCBs Report

Table 153. Fields in the Dispatcher TCBs Report

Field Heading	Description
Dispatcher TCBs	
Dispatcher Start Time and Date	The local time and date at which the CICS dispatcher started. Source field: DSGLSTRT
Address Space Accumulated CPU Time	The accumulated CPU time for this CICS address space. **Note:** This field is not reset at CICS statistics intervals. Source field: MVS field ASCBEJST
Address Space Accumulated SRB Time	The accumulated SRB time for this CICS address space. **Note:** This field is not reset at CICS statistics intervals. Source field: MVS field ASCBSRBT
Address Space CPU Time (Since Reset)	The accumulated CPU time for this CICS address space. Source field: DSGEJST
Address Space SRB Time (Since Reset)	The accumulated SRB time for this CICS address space. Source field: DSGSRBT

Table 153. Fields in the Dispatcher TCBs Report (continued)

Field Heading	Description
Number of CICS TCB modes	The number of CICS TCB modes. Source field: DSGASIZE
TCB Mode	The name of the TCB mode that the statistics refer to. The names of the TCB modes are 'QR', 'RO', 'CO', 'SZ', 'RP', 'FO', 'SL', 'SO', 'J8', 'L8', and 'S8'. Source field: DSGTCBNM
Current TCBs	The current number of TCBs attached in this mode. Source field: DSGCTCBCA
Peak TCBs	The peak number of TCBs attached in this mode. Source field: DSGTCBPA
TCBs Attached	The number of TCBs attached in this mode. Source field: DSGNTCBA
TCBs Detached - Unclean	The number of MVS TCBs that have been or are in the process of being detached for this CICS dispatcher mode because the CICS transaction associated with the TCB has abended. Source field: DSGTCBDU
TCBs Detached - Stolen	The number of MVS TCBs that have been or are in the process of being stolen from this CICS dispatcher mode because it is required by another TCB mode. Source field: DSGTCBDS
TCBs Detached - Other	The number of MVS TCBs that have been or are in the process of being detached from this CICS dispatcher TCB mode. For example, MAXOPENTCBS has been lowered, or there are too many TCBs attached in relation to the number of TCBs in use. Source field: DSGTCBDO
TCB Steals	The number of MVS TCBs that have been stolen from other TCB modes. Source field: DSGTCBST
TCB Mode	The name of the TCB mode that the statistics refer to. The names of the TCB modes are 'QR', 'RO', 'CO', 'SZ', 'RP', 'FO', 'SL', 'SO', 'J8', 'L8', and 'S8'. Source field: DSGTCBNM
Current TCBs	The current number of TCBs attached in this mode. Source field: DSGTCBCA
Peak TCBs	The peak number of TCBs attached in this mode. Source field: DSGTCBPA
Op. System Waits	The number of MVS waits which occurred on this TCB. Source field: DSGSYSW

Table 153. Fields in the Dispatcher TCBs Report (continued)

Field Heading	Description
Op. System Wait Time	The accumulated real time that this TCB was in an MVS wait, that is, the total time used between an MVS wait issued by the dispatcher and the return from the MVS wait. Source field: DSGTWT
Total TCB Dispatch Time	The accumulated real time that this TCB has been dispatched by MVS, that is, the total time used between an MVS wait issued by the dispatcher and the subsequent wait issued by the dispatcher. Source field: DSGTDT
Total TCB CPU Time	The accumulated CPU time taken for this TCB, that is, the total time that this TCB has been in execution. Source field: DSGACT
DS TCB CPU Time	The accumulated CPU time taken for this DS task, that is, the processor time used by this TCB while executing the default dispatcher task (DSTCB). Source field: DSGTCT
TCB CPU/Disp Ratio	The ratio (expressed as a percentage) of the accumulated CPU time to accumulated dispatch time for this TCB. Source field: ((DSGACT / DSGTDT) * 100)
Totals	
Current TCBs	The total number of TCBs attached for all modes. Source field: DSGTCBCA for each TCB mode
TCB Dispatch Time	The total accumulated real time that the active TCBs have been dispatched. Source field: DSGTDT for each TCB mode
TCB CPU Time	The total accumulated CPU time taken for the active TCBs. Source field: DSGACT for each TCB mode
DS TCB CPU Time	The total accumulated CPU time taken for the DS task on each active dispatcher TCB. Source field: DSGTCT for each TCB mode

Storage Reports

The Storage below 16MB report provides information on the use of MVS and CICS virtual storage. It contains the information you need to understand your current use of virtual storage below 16MB and helps you to verify the size values used for the CDSA, UDSA, SDSA, and RDSA and the value set for the DSA limit. Figure 39 on page 528 shows the format of the Storage Below 16MB Report. This report is produced using the EXEC CICS COLLECT STATISTICS STORAGE command. The statistics data is mapped by the DFHSMSDS DSECT. The field headings and contents are described in Table 154 on page 528.

```
Applid IYK2Z1V3  Sysid CJB3  Jobname CI07CJB3    Date 08/20/1998  Time 15:33:57              CICS 5.3.0              PAGE     5

  Region size established from REGION= parameter. . . :     131,072K

Storage BELOW 16MB

  Private Area Region size below 16Mb . . . . . . . . :       9,196K
    Max LSQA/SWA storage allocated below 16Mb (SYS) . :         504K
    Max User storage allocated below 16Mb (VIRT). . . :       5,436K
    System Use. . . . . . . . . . . . . . . . . . . . :          20K
    RTM . . . . . . . . . . . . . . . . . . . . . . . :         250K
  ____________________________________________________________________
  Private Area storage available below 16Mb . . . . . :       2,986K
                                                                                Current DSA Limit . . . . . . :       5,120K
                                                                                Current Allocation for DSAs . :       2,048K
  VIRT minus Current DSA Limit. . . . . . . . . . . . :         316K            Peak Allocation for DSAs. . . :       2,048K


                                        CDSA            UDSA            SDSA            RDSA           Totals
                                    ____________________________________________________________________________
  Current DSA Size. . . . . . . :         512K          1,024K            256K            256K          2,048K
  Current DSA Used. . . . . . . :         320K             0K             92K             208K            620K
  Current DSA Used as % of DSA. :          62%             0%             35%              81%             30% of DSA Size
* Peak DSA Used . . . . . . . . :         320K             4K             92K             208K
  Peak DSA Size . . . . . . . . :         512K          1,024K            256K            256K
  Cushion Size. . . . . . . . . :          64K             64K             64K              64K
  Free Storage (inc. Cushion) . :         192K          1,024K            164K              48K
* Peak Free Storage . . . . . . :         196K          1,024K            164K              48K
* Lowest Free Storage . . . . . :         192K          1,020K            164K              48K
  Largest Free Area . . . . . . :         140K          1,024K            164K              48K
  Largest Free Area as % of DSA :          27%             100%             64%              18%
  Largest Free/Free Storage . . :        0.72             1.00            1.00             1.00

  Current number of extents . . :           2                1               1                1               5
  Number of extents added . . . :           0                1               0                0               1
  Number of extents released. . :           0                0               0                0               0

  Getmain Requests. . . . . . . :          19                3               0                0
  Freemain Requests . . . . . . :          17                3               0                0

  Current number of Subpools. . :          27                9               7                3              46
  Add Subpool Requests. . . . . :          10               10               0                0
  Delete Subpool Requests . . . :           9                9               0                0

  Times no storage returned . . :           0                0               0                0
  Times request suspended . . . :           0                0               0                0
  Current requests suspended. . :           0                0               0                0
  Peak requests suspended . . . :           0                0               0                0
  Requests purged while waiting :           0                0               0                0

  Times Cushion released. . . . :           0                0               0                0               0
  Times Short-On-Storage. . . . :           0                0               0                0               0
  Total time Short-On-Storage . : 00:00:00.00000   00:00:00.00000  00:00:00.00000  00:00:00.00000
  Average Short-On-Storage time : 00:00:00.00000   00:00:00.00000  00:00:00.00000  00:00:00.00000

  Storage Violations. . . . . . :           0                0               0                0               0

  Access. . . . . . . . . . . . :        CICS             USER            USER         READONLY

  '*' indicates values reset on last DSA Size change
```

Figure 39. The Storage Report BELOW 16MB

Table 154. Fields in the Storage Below 16MB Report

Field Heading	Description
Region size established from REGION= parameter	The region size established from the REGION= parameter in the JCL. If the region requested was greater than 16 megabytes, the region established resides above 16 megabytes, and this field will be a minimum value of 32 megabytes.
Storage BELOW 16MB	
Private Area Region size below 16MB	The private area size below 16MB expressed in Kbytes.
Max LSQA/SWA storage allocated below 16MB (SYS)	The maximum amount of virtual storage allocated from the local system queue area (LSQA) and the SWA subpools below 16MB expressed in Kbytes.

Table 154. Fields in the Storage Below 16MB Report (continued)

Field Heading	Description
Max User storage allocated below 16MB (VIRT)	The maximum amount of virtual storage allocated from the user subpools below 16MB expressed in Kbytes.
System Use	is an amount of virtual storage available for system use.
RTM	is an amount of virtual storage available for use by the MVS recovery and termination manager included for calculation purposes, which could be allocated during a CICS region recovery and termination.
Private Area Storage available below 16MB	The amount of additional storage that could be allocated by increasing the REGION parameter.
Current DSA Limit	The current DSA Limit, expressed in Kbytes. Source field: (SMSDSALIMIT / 1024)
Current Allocation for DSAs	The current amount of storage allocated to the DSAs below 16MB, expressed in Kbytes. This value may be smaller or larger than the current DSA limit. Source field: (SMSDSATOTAL / 1024)
VIRT minus Current DSA Limit	The total amount of user storage allocated/used below 16MB minus the current DSA limit. Source field: ((VIRT - SMSDSALIMIT) / 1024)
Peak Allocation for DSAs	The peak amount of storage allocated to the DSAs below 16MB, expressed in Kbytes. This value may be smaller or larger than the current DSA limit. Source field: (SMSHWMDSATOTAL / 1024)
Current DSA Size	The current size of the CDSA, UDSA, SDSA, or the RDSA, expressed in Kbytes. Source field: (SMSDSASZ / 1024)
Current DSA Used	The current amount of storage used in this DSA expressed in Kbytes. Source field: ((SMSDSASZ - SMSFSTG) / 1024)
Current DSA Used as % of DSA	The current amount of storage used in this DSA expressed as a percentage of the current DSA size. Source field: (((SMSDSASZ - SMSFSTG) / SMSDSASZ) * 100)
Peak DSA Used	The peak amount of storage used in this DSA expressed in Kbytes. Source field: (SMSHWMPS / 1024)
Peak DSA Size	The peak size of the CDSA, UDSA, SDSA, or the RDSA, expressed in Kbytes. Source field: (SMSHWMDSASZ / 1024)
Cushion Size	The size of the cushion, expressed in bytes. The cushion forms part of the CDSA, UDSA, SDSA, or the RDSA, and is the amount of storage below which CICS goes SOS. Source field: (SMSCSIZE / 1024)

Table 154. Fields in the Storage Below 16MB Report (continued)

Field Heading	Description
Free Storage (inc. Cushion)	The current amount of free storage in this pagepool, expressed in Kbytes. Source field: (SMSFSTG / 1024)
Peak Free Storage	The peak amount of free storage in this pagepool, expressed in Kbytes. Source field: (SMSHWMFSTG / 1024)
Lowest Free Storage	The lowest amount of free storage in this pagepool, expressed in Kbytes. Source field: (SMSLWMFSTG / 1024)
Largest Free Area	The length of the largest contiguous free area in the CDSA, UDSA, SDSA, or RDSA, expressed in bytes. Source field: (SMSLFA / 1024)
Largest Frea Area as % of DSA	The largest contiguous free area in the CDSA, UDSA, SDSA, or RDSA, expressed as a precentage of the current DSA Size. Source field: ((SMSLFA / SMSDSASZ) * 100)
Largest Frea/Free Storage	is an indication of the storage fragmentation in this pagepool. This value is calculated by dividing the "Largest Free Area" (SMSLFA) by the "Free storage" (SMSFSTG). If the ratio is large, this pagepool is fragmented. Source field: (SMSLFA / SMSFSTG)
Current number of extents	The current number of extents allocated to this DSA. Source field: SMSEXTS
Number of extents added	The number of extents added to this DSA. Source field: SMSEXTSA
Number of extents released	The number of extents released from this DSA. Source field: SMSEXTSR
Getmain Requests	The number of GETMAIN requests from the CDSA, UDSA, SDSA, or RDSA. Source field: SMSGMREQ
Freemain Requests	The number of FREEMAIN requests from the CDSA, UDSA, SDSA, or RDSA. Source field: SMSFMREQ
Current number of Subpools	The current number of subpools (domain and task) in the CDSA, UDSA, SDSA, or RDSA. Source field: SMSCSUBP
Add Subpool Requests	The number of ADD_SUBPOOL requests to create a subpool (domain or task) from the CDSA, UDSA, SDSA, or RDSA. Source field: SMSASR

Table 154. Fields in the Storage Below 16MB Report (continued)

Field Heading	Description
Delete Subpool Requests	The number of DELETE_SUBPOOL requests (domain or task) from the CDSA, UDSA, SDSA, or RDSA. Source field: SMSDSR
Times no storage returned	The number of times a GETMAIN request with SUSPEND(NO) returned the condition INSUFFICIENT_STORAGE. Source field: SMSCRISS
Times request suspended	The number of times a GETMAIN request with SUSPEND(YES) was suspended because of insufficient storage to satisfy the request at the moment. Source field: SMSUCSS
Current requests suspended	The number of GETMAIN requests currently suspended for storage. Source field: SMSCSS
Peak requests suspended	The peak number of GETMAIN requests suspended for storage. Source field: SMSHWMSS
Requests purged while waiting	The number of requests which were purged while suspended for storage. Source field: SMSPWWS
Times cushion released	The number of times a GETMAIN request caused the storage cushion to be released. The cushion is said to be released when the number of free pages drops below the number of pages in the cushion. Source field: SMSCREL
Times Short-On-Storage	The number of times CICS went SOS in this pagepool (CDSA, UDSA, SDSA, or RDSA), where SOS means either that the cushion is currently in use and/or there is at least one task suspended for storage. Source field: SMSSOS
Total Short-On-Storage time	The accumulated time that CICS has been SOS in this DSA. Source field: SMSTSOS
Average Short-On-Storage time	The average time that CICS has been SOS in this DSA. Source field: (SMSTSOS / SMSSOS)
Storage Violations	The number of storage violations recorded in the CDSA, UDSA, SDSA, or the RDSA. Source field: SMSSV

Table 154. Fields in the Storage Below 16MB Report (continued)

Field Heading	Description
Access	The type of access of the page pool. It will either be CICS, USER, or READONLY. If storage protection is not active, all storage areas will revert to CICS except those in the RDSA. • CICS - access is CICS key • USER - access is USER key • READONLY - access is read-only protection. Source field: SMSACCESS

The Storage Above 16MB Report provides information on the use of MVS and CICS virtual storage. It contains the information you need to understand your current use of virtual storage above 16MB and helps you to verify the size values used for the ECDSA, EUDSA, ESDSA, and ERDSA and the value set for the EDSA limit. Figure 40 on page 533 shows the format of the Storage Above 16MB Report. This report is produced using the EXEC CICS COLLECT STATISTICS STORAGE command. The statistics data is mapped by the DFHSMSDS DSECT. The field headings and contents are described in Table 155 on page 533.

```
Applid IYK2Z1V3  Sysid CJB3  Jobname CI07CJB3    Date 08/20/1998  Time 15:33:57                CICS 5.3.0                PAGE     6

Storage ABOVE 16MB

  Private Area Region size above 16Mb . . . . . . . . . :   1,837,056K
    Max LSQA/SWA storage allocated above 16Mb (SYS) . :      10,084K
    Max User storage allocated above 16Mb (EXT) . . . :      71,704K
  ____________________________________________________________________
  Private Area storage available above 16Mb . . . . . :   1,755,268K
                                                                                  Current EDSA Limit. . . . . . :      65,536K
  CICS Trace table size . . . . . . . . . . . . . . . :       2,000K             Current Allocation for EDSAs. :      17,408K
  EXT minus Current EDSA Limit. . . . . . . . . . . . :       6,168K             Peak Allocation for EDSAs . . :      17,408K


                                        ECDSA            EUDSA            ESDSA            ERDSA            Totals
                                    _____________    _____________    _____________    _____________    _____________
  Current DSA Size. . . . . . . :         5,120K           1,024K           1,024K          10,240K          17,408K
  Current DSA Used. . . . . . . :         4,984K           1,024K              12K           9,648K          15,668K
  Current DSA Used as % of DSA. :            97%              100%               1%              94%              90% of EDSA Size
* Peak DSA Used . . . . . . . . :         5,044K           1,024K              12K           9,648K
  Peak DSA Size . . . . . . . . :         5,120K           1,024K           1,024K          10,240K
  Cushion Size. . . . . . . . . :           128K               0K             128K             256K
  Free Storage (inc. Cushion) . :           136K               0K           1,012K             592K
* Peak Free Storage . . . . . . :           292K           1,024K           1,024K             920K
* Lowest Free Storage . . . . . :            76K               0K           1,012K             592K
  Largest Free Area . . . . . . :           116K               0K           1,012K             452K
  Largest Free Area as % of DSA :             2%               0%              98%               4%
  Largest Free/Free Storage . . :          0.85             0.00             1.00             0.76

  Current number of extents . . :             5                1                1                9               16
  Number of extents added . . . :             0                0                1                0                1
  Number of extents released. . :             0                0                0                0                0

  Getmain Requests. . . . . . . :         1,618               24                3                4
  Freemain Requests . . . . . . :         1,503               23                0                0

  Current number of Subpools. . :           236                9                4                3              252
  Add Subpool Requests. . . . . :            12               10                0                0
  Delete Subpool Requests . . . :             9                9                0                0

  Times no storage returned . . :             0                0                0                0
  Times request suspended . . . :             0                0                0                0
  Current requests suspended. . :             0                0                0                0
  Peak requests suspended . . . :             0                0                0                0
  Requests purged while waiting :             0                0                0                0

  Times Cushion released. . . . :             0                0                0                0                0
  Times Short-On-Storage. . . . :             0                0                0                0                0
  Total time Short-On-Storage . : 00:00:00.00000   00:00:00.00000   00:00:00.00000   00:00:00.00000
  Average Short-On-Storage time : 00:00:00.00000   00:00:00.00000   00:00:00.00000   00:00:00.00000

  Storage Violations. . . . . . :             0                0                0                0                0

  Access. . . . . . . . . . . . :          CICS             USER             USER         READONLY

  '*' indicates values reset on last DSA Size change
```

Figure 40. The Storage Report ABOVE 16MB

Table 155. Fields in the Storage Above 16MB Report

Field Heading	Description
Storage ABOVE 16MB	
Private Area Region size above 16MB	The private area size above 16MB expressed in Kbytes.
Max LSQA/SWA storage allocated above 16MB (SYS)	The maximum amount of virtual storage allocated from the local system queue area (LSQA) and the SWA subpools above 16MB expressed in Kbytes.
Max User storage allocated above 16MB (EXT)	The maximum amount of virtual storage allocated from the user subpools above 16MB expressed in Kbytes.
Private Area Storage available above 16MB	The amount of additional storage that could be allocated by increasing the REGION parameter.
Current EDSA Limit	The current EDSA Limit, expressed in Kbytes. Source field: (SMSEDSALIMIT / 1024)

Table 155. Fields in the Storage Above 16MB Report (continued)

Field Heading	Description
CICS Trace table size	The current size of the CICS trace table. Source field: EXEC CICS INQUIRE TRACEDEST TABLESIZE
Current Allocation for EDSAs	The peak amount of storage allocated to the DSAs above 16MB, expressed in Kbytes. This value may be smaller or larger than the current EDSA limit. Source field: (SMSEDSATOTAL / 1024)
EXT minus Current EDSA Limit	The total amount of user storage allocated/used above 16MB minus the current DSA limit. Source field: ((EXT - SMSEDSALIMIT) / 1024)
Peak Allocation for EDSAs	The peak amount of storage allocated to the DSAs above 16MB, expressed in Kbytes. This value may be smaller or larger than the current EDSA limit. Source field: (SMSHWMEDSATOTAL / 1024)
Current DSA Size	The current size of the ECDSA, EUDSA, ESDSA, or the ERDSA, expressed in Kbytes. Source field: (SMSDSASZ / 1024)
Current DSA Used	The current amount of storage used in this DSA expressed in Kbytes. Source field: ((SMSDSASZ - SMSFSTG) / 1024)
Current DSA Used as % of DSA	The current amount of storage used in this DSA expressed as a percentage of the current DSA size. Source field: (((SMSDSASZ - SMSFSTG) / SMSDSASZ) * 100)
Peak DSA Used	The peak amount of storage used in this DSA expressed in Kbytes. Source field: (SMSHWMPS / 1024)
Peak DSA Size	The peak size of the ECDSA, EUDSA, ESDSA, or the ERDSA, expressed in Kbytes. Source field: (SMSHWMDSASZ / 1024)
Cushion Size	The size of the cushion, expressed in bytes. The cushion forms part of the ECDSA, EUDSA, ESDSA, or the ERDSA, and is the amount of storage below which CICS goes SOS. Source field: (SMSCSIZE / 1024)
Free Storage (inc. Cushion)	The current amount of free storage in this pagepool, expressed in Kbytes. Source field: (SMSFSTG / 1024)
Peak Free Storage	The peak amount of free storage in this pagepool, expressed in Kbytes. Source field: (SMSHWMFSTG / 1024)

Table 155. Fields in the Storage Above 16MB Report (continued)

Field Heading	Description
Lowest Free Storage	The lowest amount of free storage in this pagepool, expressed in Kbytes. Source field: (SMSLWMFSTG / 1024)
Largest Free Area	The length of the largest contiguous free area in the ECDSA, EUDSA, ESDSA, or ERDSA, expressed in Kbytes. Source field: SMSLFA
Largest Frea Area as % of DSA	The largest contiguous free area in the ECDSA, EUDSA, ESDSA, or ERDSA, expressed as a percentage of the current DSA Size. Source field: ((SMSLFA / SMSDSASZ) * 100)
Largest Frea/Free Storage	is an indication of the storage fragmentation in this pagepool. This value is calculated by dividing the "Largest Free Area" (SMSLFA) by the "Free storage" (SMSFSTG). If the ratio is large, this pagepool is fragmented. Source field: (SMSLFA / SMSFSTG)
Current number of extents	The current number of extents allocated to this DSA. Source field: SMSEXTS
Number of extents added	The number of extents added to this DSA. Source field: SMSEXTSA
Number of extents released	The number of extents released from this DSA. Source field: SMSEXTSR
Getmain Requests	The number of GETMAIN requests from the ECDSA, EUDSA, ESDSA, or ERDSA. Source field: SMSGMREQ
Freemain Requests	The number of FREEMAIN requests from the ECDSA, EUDSA, ESDSA, or ERDSA. Source field: SMSFMREQ
Current number of Subpools	The current number of subpools (domain and task) in the ECDSA, EUDSA, ESDSA, or ERDSA. Source field: SMSCSUBP
Add Subpool Requests	The number of ADD_SUBPOOL requests to create a subpool (domain or task) from the ECDSA, EUDSA, ESDSA, or ERDSA. Source field: SMSASR
Delete Subpool Requests	The number of DELETE_SUBPOOL requests (domain or task) from the ECDSA, EUDSA, ESDSA, or ERDSA. Source field: SMSDSR

Table 155. Fields in the Storage Above 16MB Report (continued)

Field Heading	Description
Times no storage returned	The number of times a GETMAIN request with SUSPEND(NO) returned the condition INSUFFICIENT_STORAGE. Source field: SMSCRISS
Times request suspended	The number of times a GETMAIN request with SUSPEND(YES) was suspended because of insufficient storage to satisfy the request at the moment. Source field: SMSUCSS
Current requests suspended	The number of GETMAIN requests currently suspended for storage. Source field: SMSCSS
Peak requests suspended	The peak number of GETMAIN requests suspended for storage. Source field: SMSHWMSS
Requests purged while waiting	The number of requests which were purged while suspended for storage. Source field: SMSPWWS
Times cushion released	The number of times a GETMAIN request caused the storage cushion to be released. The cushion is said to be released when the number of free pages drops below the number of pages in the cushion. Source field: SMSCREL
Times Short-On-Storage	The number of times CICS went SOS in this pagepool (ECDSA, EUDSA, ESDSA, or ERDSA), where SOS means either that the cushion is currently in use and/or there is at least one task suspended for storage. Source field: SMSSOS
Total Short-On-Storage time	The accumulated time that CICS has been SOS in this DSA. Source field: SMSTSOS
Average Short-On-Storage time	The average time that CICS has been SOS in this DSA. Source field: (SMSTSOS / SMSSOS)
Storage Violations	The number of storage violations recorded in the ECDSA, EUDSA, ESDSA, or the ERDSA. Source field: SMSSV
Access	The type of access of the page pool. It will either be CICS, USER, or READONLY. If storage protection is not active, all storage areas will revert to CICS except those in the ERDSA. • CICS - access is CICS key • USER - access is USER key • READONLY - access is read-only protection Source field: SMSACCESS

Loader and Program Storage Report

Figure 41 shows the format of the Loader and Program Storage Report. This report is produced using a combination of the EXEC CICS COLLECT STATISTICS PROGRAM and EXEC CICS COLLECT STATISTICS STORAGE commands. The statistics data is mapped by the DFHLDGDS and DFHSMDDS DSECTs. The field headings and contents are described in Table 156.

```
Applid IYK2Z1V3  Sysid CJB3  Jobname CI07CJB3    Date 08/20/1998  Time 15:33:57                CICS 5.3.0                PAGE     7


Loader

  Library Load requests. . . . . . . . . . . . . :             15  Total Program Uses . . . . . . . . . . . . . . :            541
  Total Library Load time. . . . . . . . . . . . :  00:00:00.34448  Program Use to Load Ratio. . . . . . . . . . . :          36.06
  Average Library Load time. . . . . . . . . . . :  00:00:00.02296
                                                                    Times DFHRPL secondary extents detected. . . . :              0
  Library Load requests that waited. . . . . . . :              0
  Total Library Load request wait time . . . . . :  00:00:00.00000
  Average Library Load request wait time . . . . :  00:00:00.00000
  Current Waiting Library Load requests. . . . . :              0
  Peak Waiting Library Load requests . . . . . . :              0
  Times at Peak. . . . . . . . . . . . . . . . . :              0  Average Not-In-Use program size. . . . . . . . :            14K


  CDSA                                                              ECDSA

  Programs Removed by compression. . . . . . . . :              0  Programs Removed by compression. . . . . . . . :              0
  Time on the Not-In-Use Queue . . . . . . . . . :  00:00:00.00000  Time on the Not-In-Use Queue . . . . . . . . . :  00:00:00.00000
  Average Time on the Not-In-Use Queue . . . . . :  00:00:00.00000  Average Time on the Not-In-Use Queue . . . . . :  00:00:00.00000
  Programs Reclaimed from the Not-In-Use Queue . :              0  Programs Reclaimed from the Not-In-Use Queue . :            435
  Programs Loaded - now on the Not-In-Use Queue. :              0  Programs Loaded - now on the Not-In-Use Queue. :             15

  SDSA                                                              ESDSA

  Programs Removed by compression. . . . . . . . :              0  Programs Removed by compression. . . . . . . . :              0
  Time on the Not-In-Use Queue . . . . . . . . . :  00:00:00.00000  Time on the Not-In-Use Queue . . . . . . . . . :  00:00:00.00000
  Average Time on the Not-In-Use Queue . . . . . :  00:00:00.00000  Average Time on the Not-In-Use Queue . . . . . :  00:00:00.00000
  Programs Reclaimed from the Not-In-Use Queue . :              0  Programs Reclaimed from the Not-In-Use Queue . :              2
  Programs Loaded - now on the Not-In-Use Queue. :              0  Programs Loaded - now on the Not-In-Use Queue. :              3

  RDSA                                                              ERDSA

  Programs Removed by compression. . . . . . . . :              0  Programs Removed by compression. . . . . . . . :              0
  Time on the Not-In-Use Queue . . . . . . . . . :  00:00:00.00000  Time on the Not-In-Use Queue . . . . . . . . . :  00:00:00.00000
  Average Time on the Not-In-Use Queue . . . . . :  00:00:00.00000  Average Time on the Not-In-Use Queue . . . . . :  00:00:00.00000
  Programs Reclaimed from the Not-In-Use Queue . :              0  Programs Reclaimed from the Not-In-Use Queue . :             75
  Programs Loaded - now on the Not-In-Use Queue. :              1  Programs Loaded - now on the Not-In-Use Queue. :             24


Program Storage

  Nucleus Program Storage (CDSA) . . . . . . . . :            36K  Nucleus Program Storage (ECDSA). . . . . . . . :           104K

  Program Storage (SDSA) . . . . . . . . . . . . :             0K  Program Storage (ESDSA). . . . . . . . . . . . :            12K
  Resident Program Storage (SDSA). . . . . . . . :             0K  Resident Program Storage (ESDSA) . . . . . . . :             0K

  Read-Only Nucleus Program Storage (RDSA) . . . :          156K  Read-Only Nucleus Program Storage (ERDSA). . . :         5,988K
  Read-Only Program Storage (RDSA) . . . . . . . :            52K  Read-Only Program Storage (ERDSA). . . . . . . :         3,660K
  Read-Only Resident Program Storage (RDSA). . . :             0K  Read-Only Resident Program Storage (ERDSA) . . :             0K

  CDSA used by Not-In-Use programs. :         0K   0.00% of CDSA   ECDSA used by Not-In-Use programs :         55K   1.08% of ECDSA
  SDSA used by Not-In-Use programs. :         0K   0.00% of SDSA   ESDSA used by Not-In-Use programs :         10K   1.00% of ESDSA
  RDSA used by Not-In-Use programs. :         1K   0.22% of RDSA   ERDSA used by Not-In-Use programs :        573K   5.60% of ERDSA
```

Figure 41. The Loader and Program Storage Report

Table 156. Fields in the Loader and Program Storage Report

Field Heading	Description
Loader	
Library Load requests	The number of times the loader has issued an MVS LOAD request to load programs from the DFHRPL library concatenation into CICS managed storage. Modules in the LPA are not included in this figure. Source field: LDGLLR

Table 156. Fields in the Loader and Program Storage Report (continued)

Field Heading	Description
Total Program Uses	The number of uses of any program by the CICS system. Source field: LDGPUSES
Total Library Load time	The time taken for the number of library loads indicated by LDGLLR. Source field: LDGLLT
Program Use to Load Ratio	The ratio of program uses to programs loads. Source field: (LDGPUSES / LDGLLR)
Average Library Load time	The average time to load a program. Source field: (LDGLLT / LDGLLR)
Times DFHRPL secondary extents detected	The number of times the loader received an end-of-extent condition during a LOAD and successfully closed and reopened the DFHRPL library and retried the LOAD. Source field: LDGDREBS
Library Load requests that waited	The number of loader domain requests that ***were*** forced to suspend due to the loader domain performing an operation on that program on behalf of another task. These operations could be: • A NEWCOPY request • Searching the LPA • A physical load in progress. This figure is the total number of tasks that have waited, and does not include those that are currently waiting (LDGWLR). Source field: LDGWTDLR
Total Library Load request wait time	The suspended time for the number of tasks indicated by LDGWTDLR. Source field: LDGTTW
Average Library Load request wait time	The average loader domain request suspend time. Source field: (LDGTTW / LDGWTDLR)
Current Waiting Library Load requests	The number of loader domain requests that ***are currently*** forced to suspend due to the loader domain currently performing an operation on that program on behalf of another task. These operations could be: • A NEWCOPY request • Searching the LPA • A physical load in progress. Source field: LDGWLR
Peak Waiting Library Load requests	The maximum number of tasks suspended at one time. Source field: LDGWLRHW

Table 156. Fields in the Loader and Program Storage Report (continued)

Field Heading	Description
Times at Peak	The number of times the high watermark level indicated by LDGWLRHW was reached. This, along with the previous two values, is an indication of the level of contention for loader resource. Source field: LDGHWMT
Average Not-In-Use program size	The average size of a program currently on the Not-In-Use queue. Source field: ((LDGCNIU + LDGSNIU + LDGRNIUS + LDGECNIU + LDGESNIU + LDGERNIU) / 1024) / LDGPROGNIU)
Programs Removed by compression	The number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism. Source field: LDGDPSCR
Time on the Not-In-Use Queue	The program Not-In-Use (NIU) queue membership time. For each program that becomes eligible for removal from storage by the DPSC mechanism, the time between the program becoming eligible and the actual time of its being removed from storage is calculated. This field is the sum of these times for all programs removed by the DPSC mechanism and as such can be greater than the elapsed time CICS run time. This field does not include the wait time for those programs reclaimed from the Not-In-Use queue. Source field: LDGDPSCT
Average Time on the Not-In-Use Queue	The average length of time that a program is eligible for removal from storage by the DPSC mechanism. Source field: (LDGDPSCT / LDGDPSCR)
Programs Reclaimed from the Not-In-Use Queue	The number of reclaims that CICS has made from the Not-In-Use (NIU) queue. Reclaims occur when a request is issued for programs currently in the Not-In-Use queue. The reclaimed instance of a program is no longer eligible for program compression (DPSC). Source field: LDGRECNIU
Programs Loaded - on the Not-In-Use Queue	The number of programs on the Not-In-Use (NIU) queue. Source field: LDGPROGNIU
Program Storage	
Nucleus Program Storage (CDSA)	The current amount of storage allocated to nucleus programs in the CDSA. Source field: (SMDCPS for subpool ''LDNUC ' and 'LDNRS / 1024)

Table 156. Fields in the Loader and Program Storage Report (continued)

Field Heading	Description
Nucleus Program Storage (ECDSA)	The current amount of storage allocated to nucleus programs in the ECDSA. Source field: (SMDCPS for subpools 'LDENUC ' and 'LDENRS ' / 1024)
Program Storage (SDSA)	The current amount of storage allocated to programs in the SDSA. Source field: (SMDCPS for subpool 'LDPGM ' / 1024)
Program Storage (ESDSA)	The current amount of storage allocated to programs in the ESDSA. Source field: (SMDCPS for subpool 'LDEPGM ' / 1024)
Resident Program Storage (SDSA)	The current amount of storage allocated to resident programs in the SDSA. Source field: (SMDCPS for subpool 'LDRES ' / 1024)
Resident Program Storage (ESDSA)	The current amount of storage allocated to resident programs in the ESDSA. Source field: (SMDCPS for subpool 'LDERES ' / 1024)
Read-Only Nucleus Program Storage (RDSA)	The current amount of storage allocated to nucleus programs in the RDSA. Source field: (SMDCPS for subpools 'LDNUCRO ' and 'LDNRSRO ' / 1024)
Read-Only Nucleus Program Storage (ERDSA)	The current amount of storage allocated to nucleus programs in the ERDSA. Source field: (SMDCPS for subpools 'LDENUCRO ' and 'LDENRSRO ' / 1024)
Read-Only Program Storage (RDSA)	The current amount of storage allocated to programs in the RDSA. Source field: (SMDCPS for subpool 'LDPGMRO ' / 1024)
Read-Only Program Storage (ERDSA)	The current amount of storage allocated to programs in the ERDSA. Source field: (SMDCPS for subpool 'LDEPGMRO ' / 1024)
Read-Only Resident Program Storage (RDSA)	The current amount of storage allocated to resident programs in the RDSA. Source field: (SMDCPS for subpool 'LDRESRO ' / 1024)
Read-Only Resident Program Storage (ERDSA)	The current amount of storage allocated to resident programs in the ERDSA. Source field: (SMDCPS for subpool 'LDERESRO ' / 1024)
CDSA used by Not-In-Use programs	The current amount of CDSA storage which is occupied by Not-In-Use (NIU) programs. Source field: (LDGSTGNIU(1) / 1024)

Table 156. Fields in the Loader and Program Storage Report (continued)

Field Heading	Description
ECDSA used by Not-In-Use programs	The current amount of ECDSA storage which is occupied by Not-In-Use (NIU) programs. Source field: (LDGSTGNIU(2) / 1024)
SDSA used by Not-In-Use programs	The current amount of UDSA storage which is occupied by Not-In-Use (NIU) programs. Source field: (LDGSTGNIU(3) / 1024)
ESDSA used by Not-In-Use programs	The current amount of EUDSA storage which is occupied by Not-In-Use (NIU) programs. Source field: (LDGSTGNIU(4) / 1024)
RDSA used by Not-In-Use programs	The current amount of UDSA storage which is occupied by Not-In-Use (NIU) programs. Source field: (LDGSTGNIU(5) / 1024)
ERDSA used by Not-In-Use programs	The current amount of ERDSA storage which is occupied by Not-In-Use (NIU) programs. Source field: (LDGSTGNIU(6) / 1024)

Storage Subpools Report

Figure 42 on page 542 shows the format of the Storage Subpools Report. This report is produced using the EXEC CICS COLLECT STATISTICS STORAGE command. The statistics data is mapped by the DFHSMDDS DSECT. The field headings and contents are described in Table 157 on page 542.

```
Applid IYK2Z1V3  Sysid CJB3  Jobname CI07CJB3    Date 11/23/1998  Time 10:02:00      CICS 5.3.0       PAGE  8


Storage Subpools


      Subpool   Subpool         Current      Peak
 Name      Location        Storage      Storage

      LDNRS     CDSA              40K          40K
      LDNRSRO   RDSA             152K         152K
      LDNUC     CDSA               4K           4K
      LDNUCRO   RDSA               8K           8K
      LDPGM     SDSA               0K           0K
      LDPGMRO   RDSA              52K          52K
      LDRES     SDSA               0K           0K
      LDRESRO   RDSA               0K           0K

      LDENRS    ECDSA             44K          44K
      LDENUC    ECDSA             76K          76K
      LDEPGM    ESDSA             24K          24K
      LDERES    ESDSA              0K           0K
      LDENRSRO  ERDSA          3,628K       3,628K
      LDENUCRO  ERDSA          2,176K       2,176K
      LDEPGMRO  ERDSA          3,664K       3,664K
      LDERESRO  ERDSA              0K           0K

      SMSHARED  CDSA               4K           4K

      SMSHRC24  CDSA               0K           0K
      SMSHRU24  SDSA              84K          84K

      SMSHRC31  ECDSA              4K           4K
      SMSHRU31  ESDSA              0K           0K

      KESTK24   CDSA             188K         188K
      KESTK24E  CDSA              36K          36K

      KESTK31   ECDSA          2,256K       2,256K
      KESTK31E  ECDSA             32K          32K

      TSMAIN    ECDSA              0K           0K
```

Figure 42. The Storage Subpools Report

Table 157. Fields in the Storage Subpools Report

Field Heading	Description
Storage Subpools	
Subpool Name	The name of the domain subpool. Source field: SMDSPN
Current Storage	The current amount of storage allocated to this domain subpool. Source field: SMDCPS
Peak Storage	The peak amount of storage allocated to this domain subpool. Source field: SMDHWMPS

Transaction Classes Report

Figure 43 shows the format of the Transaction Classes Report. This report is produced using a combination of the EXEC CICS INQUIRE TRANCLASS and EXEC CICS COLLECT STATISTICS TRANCLASS commands. The statistics data is mapped by the DFHXMCDS DSECT. The field headings and contents are described in Table 158.

```
Applid IYK2Z1V3  Sysid CJB3  Jobname CI07CJB3    Date 08/20/1998  Time 15:33:57                CICS 5.3.0                PAGE     9

Transaction Classes

   Tclass   Trans Attach Class  At     Cur   Peak  Purge    At     Cur   Peak  Accept Accept Purged Purge  Total    Avg.   Avg. Cur
    Name   in Tcl in Tcl Limit Limit Active Active Thresh Thresh Queued Queued Immed  Queued Immed  Queued Queued Que Time Que Time
  ____________________________________________________________________________________________________________________________

  DFHCOMCL      2      0    10     0      0      0      0      0      0      0      0      0      0      0      0 00:00.00 00:00.00
  DFHEDFTC      0      0    10     0      0      0      0      0      0      0      0      0      0      0      0 00:00.00 00:00.00
  DFHTCIND      0      0    10     0      0      0      0      0      0      0      0      0      0      0      0 00:00.00 00:00.00
  DFHTCL01      0      0     1     0      0      0      0      0      0      0      0      0      0      0      0 00:00.00 00:00.00
  DFHTCL02      0      0     1     0      0      0      0      0      0      0      0      0      0      0      0 00:00.00 00:00.00
  DFHTCL03      0      0     6     0      0      0      0      0      0      0      0      0      0      0      0 00:00.00 00:00.00
  DFHTCL04      0      0     1     0      0      0      0      0      0      0      0      0      0      0      0 00:00.00 00:00.00
  DFHTCL05      0      0     1     0      0      0      0      0      0      0      0      0      0      0      0 00:00.00 00:00.00
  DFHTCL06      0      0     1     0      0      0      0      0      0      0      0      0      0      0      0 00:00.00 00:00.00
  DFHTCL07      0      0     1     0      0      0      0      0      0      0      0      0      0      0      0 00:00.00 00:00.00
  DFHTCL08      0      0     1     0      0      0      0      0      0      0      0      0      0      0      0 00:00.00 00:00.00
  DFHTCL09      0      0     1     0      0      0      0      0      0      0      0      0      0      0      0 00:00.00 00:00.00
  DFHTCL10      0      0     1     0      0      0      0      0      0      0      0      0      0      0      0 00:00.00 00:00.00

  Totals        2      0

Transaction Classes . :          13
```

Figure 43. The Transaction Classes Report

Table 158. Fields in the Transaction Classes Report

Field Heading	Description
Transaction Classes	
Tclass Name	The name of the transaction class. Source field: EXEC CICS INQUIRE TRANCLASS()
Trans in Tcl	The number of transaction definitions that are defined to this transaction class. Source field: XMCITD
Attach in Tcl	The number of transaction attach requests for transactions in this transaction class. Source field: XMCTAT
Class Limit	The maximum number of transactions that may be concurrently active in this transaction class. Source field: XMCMXT
At Limit	The number of times that this transaction class has reached its transaction class limit. Source field: XMCTAMA
Cur Active	The current number of transactions active in this transaction class. Source field: XMCCAT

Table 158. Fields in the Transaction Classes Report (continued)

Field Heading	Description
Peak Active	The peak number of transactions active in this transaction class. Source field: XMCPAT
Purge Thresh	The queue limit purge threshold for this transaction class. Source field: XMCTH
At Thresh	The number of times this transaction class has reached its queue limit purge threshold. Source field: XMCTAPT
Cur Queued	The current number of transactions that are currently queueing in this transaction class. Source field: XMCCQT
Peak Queued	The peak number of transactions that queued waiting to get into this transaction class. Source field: XMCPQT
Accept Immed	The number of transactions that were accepted immediately into this transaction class. Source field: XMCAI
Accept Queued	The number of transactions that were queued before being accepted into this transaction class. Source field: XMCAAQ
Purged Immed	The number of transaction that were purged immediately because the queue had already reached the purge threshold for this transaction class. Source field: XMCPI
Purged Queued	The number of transaction that have been purged whilst queueing to get into this transaction class. Source field: XMCPWQ
Total Queued	The total number of transactions that have become active but first queued to get into this transaction class. Source field: XMCTQ
Avg. Que Time	The average queueing time transactions that have become active but first queued to get into this transaction class. Source field: XMCTQTME / XMCTQ
Avg. Cur Que Time	The average queueing time for those transactions that are currently queued waiting to get into this transaction class. Source field: XMCCQTME / XMCCQT

Transactions Report

Figure 44 shows the format of the Transactions report. This report is produced using a combination of the EXEC CICS INQUIRE TRANSACTION and EXEC CICS COLLECT STATISTICS TRANSACTION commands. The statistics data is mapped by the DFHXMRDS DSECT. The field headings and contents are described in Table 159.

```
Applid IYK2Z1V3  Sysid CJB3  Jobname CI07CJB3    Date 08/20/1998  Time 15:33:57                CICS 5.3.0                PAGE    10

Transactions

  Tran    Tran    Program                          Task Data      Attach      Restart     Dynamic --- Counts        Remote      Storage
   id     Class     Name    Dynamic  Isolate  Location/Key    Count       Count       Local   -  Remote        Starts       Viols

  -ALT            DFHD2CM1  Static     Yes       Any/CICS           0           0           0           0           0           0
  -ARC            DFHD2CM1  Static     Yes       Any/CICS           0           0           0           0           0           0
  -CAN            DFHD2CM1  Static     Yes       Any/CICS           0           0           0           0           0           0
  -DIS            DFHD2CM1  Static     Yes       Any/CICS           0           0           0           0           0           0
  -MOD            DFHD2CM1  Static     Yes       Any/CICS           0           0           0           0           0           0
  -REC            DFHD2CM1  Static     Yes       Any/CICS           0           0           0           0           0           0
  -RES            DFHD2CM1  Static     Yes       Any/CICS           0           0           0           0           0           0
  -SET            DFHD2CM1  Static     Yes       Any/CICS           0           0           0           0           0           0
  -STA            DFHD2CM1  Static     Yes       Any/CICS           0           0           0           0           0           0
  -STO            DFHD2CM1  Static     Yes       Any/CICS           0           0           0           0           0           0
  -TER            DFHD2CM1  Static     Yes       Any/CICS           0           0           0           0           0           0
  AADD            DFH$AALL  Static     Yes     Below/USER           0           0           0           0           0           0
  ABRW            DFH$ABRW  Static     Yes     Below/USER           0           0           0           0           0           0
  ADMA            ADMIVPC   Static     Yes     Below/USER           0           0           0           0           0           0
  ADMC            ADMPSTBC  Static     Yes     Below/USER           0           0           0           0           0           0
  ADMI            ADMISSEC  Static     Yes     Below/USER           0           0           0           0           0           0
  ADMM            ADM1IMDC  Static     Yes     Below/USER           0           0           0           0           0           0
  ADMP            ADMOPUC   Static     Yes     Below/USER           0           0           0           0           0           0
  ADMU            ADM5IVUC  Static     Yes     Below/USER           0           0           0           0           0           0
  ADMV            ADMVSSEC  Static     Yes     Below/USER           0           0           0           0           0           0
  ADM4            ADM4CDUC  Static     Yes     Below/USER           0           0           0           0           0           0
  ADYN            DFH99     Static     Yes     Below/CICS           0           0           0           0           0           0
  AINQ            DFH$AALL  Static     Yes     Below/USER           0           0           0           0           0           0
  AMNU            DFH$AMNU  Static     Yes     Below/USER           0           0           0           0           0           0
  AORD            DFH$AREN  Static     Yes     Below/USER           0           0           0           0           0           0
  AORQ            DFH$ACOM  Static     Yes     Below/USER           0           0           0           0           0           0
  APPA            APPCP05   Static     Yes       Any/USER           0           0           0           0           0           0
  APPC            APPCP00   Static     Yes       Any/USER           0           0           0           0           0           0
  AREP            DFH$AREP  Static     Yes     Below/USER           0           0           0           0           0           0
  AUPD            DFH$AALL  Static     Yes     Below/USER           0           0           0           0           0           0
  BACK            DPLBACK   Static     Yes     Below/USER           0           0           0           0           0           0
  BRAS            BRASSIGN  Static     Yes     Below/USER           0           0           0           0           0           0
  BRA1            BRA009BS  Static     Yes     Below/USER           0           0           0           0           0           0
  BRA2            BRA010BS  Static     Yes     Below/USER           0           0           0           0           0           0
  BRA5            BRA005BS  Static     Yes     Below/USER           0           0           0           0           0           0
  BRLT            BRSTLTBS  Static     Yes     Below/USER           0           0           0           0           0           0
  BRU1            BRU001BS  Static     Yes     Below/USER           0           0           0           0           0           0
  BRU2            BRU002BS  Static     Yes     Below/USER           0           0           0           0           0           0
  BRU3            BRU003BS  Static     Yes     Below/USER           0           0           0           0           0           0
  BRU4            BRU004BS  Static     Yes     Below/USER           0           0           0           0           0           0
  BRU5            BRU005BS  Static     Yes     Below/USER           0           0           0           0           0           0
  BRU6            BRU006BS  Static     Yes     Below/USER           0           0           0           0           0           0
  CAFB            CAUCAFB1  Static     Yes       Any/CICS           0           0           0           0           0           0
  CAFF            CAUCAFF1  Static     Yes       Any/CICS           0           0           0           0           0           0
  CALL            CALLJT1   Static     Yes       Any/USER           0           0           0           0           0           0
  CATA            DFHZATA   Static     Yes       Any/CICS           0           0           0           0           0           0
  CATD            DFHZATD   Static     Yes       Any/CICS           0           0           0           0           0           0
  CATR            DFHZATR   Static     Yes       Any/CICS           0           0           0           0           0           0
  CBAM            DFHECBAM  Static     Yes     Below/CICS           0           0           0           0           0           0
  CBLT            DFHDUMMY  Static     Yes       Any/CICS           0           0           0           0           0           0
  CCIN  DFHCOMCL  DFHZCN1   Static     Yes       Any/CICS           0           0           0           0           0           0
```

Figure 44. The Transactions Report

Table 159. Fields in the Transactions Report

Field Heading	Description
Transactions	
Tran id	The name of the transaction. Source field: EXEC CICS INQUIRE TRANSACTION

Table 159. Fields in the Transactions Report (continued)

Field Heading	Description
Tran Class	The name of the transaction class in which the transaction is defined. Source field: XMRTCL
Program Name	The name of the program when the transaction was defined, or spaces if a program name was not supplied. Source field: XMMRPN
Dynamic	Indicates whether the transaction was defined as dynamic. Source field: XMRDYN
Isolate	Indicates whether the transaction's user-key task-lifetime storage is isolated from the user-key programs of other transactions. Source field: EXEC CICS INQUIRE TRANSACTION ISOLATEST
Task Data Location	Where certain CICS control blocks will be located for the transaction. Source field: EXEC CICS INQUIRE TRANSACTION TASKDATALOC
Task Data Key	The storage key in which CICS will obtain all storage for use by the transaction. Source field: EXEC CICS INQUIRE TRANSACTION TASKDATAKEY
Attach Count	The number of times this transaction definition has been used to attach a transaction. Source field: XMRAC
Restart Count	The number of times this transaction was restarted after an abend. This field is zero if the transaction was not defined as RESTART=YES. Source field: XMRRC
Dynamic Counts - Local	The total number of times the dynamic transaction routing exit has chosen to run this transaction on the local system. This field is zero if the transaction was not defined as DYNAMIC=YES. Source field: XMRDLC
Dynamic Counts - Remote	The total number of times the dynamic transaction routing exit has chosen to run this transaction on a remote system. This field is zero if the transaction was not defined as DYNAMIC=YES. Source field: XMRDRC
Remote Starts	The total number of times this transaction definition has been used to start a transaction remotely. Source field: XMRRSC
Storage Viols	Source field: XMRSVC

Transaction Totals Report

Figure 45 on page 547 shows the format of the Transactions Totals Report. The field headings and contents are described in Table 160 on page 547.

```
Applid IYK2Z1V3  Sysid CJB3  Jobname CI07CJB3    Date 08/20/1998  Time 15:33:57              CICS 5.3.0              PAGE    15


Transaction Totals

            Task Data    Subspace  Transaction    Attach
  Isolate  Location/Key   Usage       Count       Count

     Yes     Below/CICS     None           34           4
     Yes       Any/CICS     None          106           3
     Yes     Below/USER    Unique          53           0
     Yes       Any/USER    Unique          45           2

      No     Below/CICS    Common           0           0
      No       Any/CICS    Common           1           0
      No     Below/USER    Common           3           0
      No       Any/USER    Common           0           0
Totals                                    242           9


Subspace Statistics

  Current Unique Subspace Users (Isolate=Yes). . . :          0
  Peak Unique Subspace Users (Isolate=Yes) . . . . :          1
  Total Unique Subspace Users (Isolate=Yes). . . . :          2

  Current Common Subspace Users (Isolate=No) . . . :          0
  Peak Common Subspace Users (Isolate=No). . . . . :          0
  Total Common Subspace Users (Isolate=No) . . . . :          0
```

Figure 45. The Transaction Totals Report

Table 160. Fields in the Transaction Totals Report

Field Heading	Description
Transaction Totals	
Isolate	Indicates whether the transaction's user-key task-lifetime storage is isolated from the user-key programs of other transactions.
Task Data Location/Key	The number of transactions defined with this combination of task data location and task data key.
Subspace Usage	Indicates the type of subspace usage for these transaction definitions.
Transaction Count	The number of transaction definitions for this combination of isolate, task data location, task data key, and subspace usage.
Attach Count	The number of times these transaction definitions have been used to attach a transaction.
Subspace Statistics	
Current Unique Subspace Users (Isolate=Yes)	The current number of tasks allocated a unique subspace. Source field: SMSUSSCUR
Peak Unique Subspace Users (Isolate=Yes)	The peak number of tasks allocated a unique subspace. Source field: SMSUSSHWM
Total Unique Subspace Users (Isolate=Yes)	The total number of tasks that have been allocated a unique subspace. Source field: SMSUSSCUM
Current Common Subspace Users (Isolate=No)	The current number of tasks allocated to the common subspace. Source field: SMSCSSCUR

Table 160. Fields in the Transaction Totals Report (continued)

Field Heading	Description
Peak Common Subspace Users (Isolate=No)	The peak number of tasks allocated to the common subspace. Source field: SMSCSSHWM
Total Common Subspace Users (Isolate=No)	The total number of tasks that have been allocated to the common subspace. Source field: SMSCSSCUM

Programs Report

Figure 46 on page 549 shows the format of the Programs Report. This report is produced using a combination of the EXEC CICS INQUIRE PROGRAM and EXEC CICS COLLECT STATISTICS PROGRAM commands. The statistics data was mapped by the DFHLDRDS DSECT. The field headings and contents are described in Table 161 on page 549.

Programs

Program Name	Data Loc	Exec Key	Times Used	Times Fetched	Total Fetch Time	Average Fetch Time	RPL Offset	Times Newcopy	Times Removed	Program Size	Program Location
DFHPGAHX	Any	CICS	0					0	0		None
DFHPGALX	Any	CICS	0					0	0		None
DFHPGAMP			0					0	0		None
DFHPGAOX	Any	CICS	0					0	0		None
DFHPGAPG	Below	USER	0					0	0		None
DFHPGAPT			0					0	0		None
DFHPRK	Any	CICS	0					0	0		None
DFHPSIP	Any	CICS	0	0			2	0	0	864	ECDSA
DFHPUP	Any	CICS	14	0			2	0	0	20,904	ERDSA
DFHP3270	Any	CICS	0					0	0		None
DFHQRY	Any	CICS	0	0			2	0	0	3,936	ERDSA
DFHRCEX	Any	CICS	0					0	0	944	None
DFHREST	Any	CICS	0					0	0		None
DFHRKB	Any	CICS	0					0	0		None
DFHRMSY	Any	CICS	0					0	0		None
DFHRMXN3	Any	CICS	0					0	0		None
DFHRPAL	Any	CICS	0					0	0		None
DFHRPAS	Any	CICS	0					0	0		None
DFHRPC00	Any	CICS	0					0	0		None
DFHRPMS	Any	CICS	0					0	0		None
DFHRPRP	Any	CICS	0					0	0		None
DFHRPTRU	Any	USER	0					0	0		None
DFHRP0			0					0	0		None
DFHRTC	Any	CICS	0					0	0		None
DFHRTE	Any	CICS	0					0	0		None
DFHSFP	Any	CICS	0					0	0		None
DFHSHRRP	Any	CICS	0					0	0		None
DFHSHRSP	Any	CICS	0					0	0		None
DFHSHSY	Any	CICS	0	0			2	0	0	632	ERDSA
DFHSIPLT	Any	CICS	0					0	0	11,152	None
DFHSNLE			0	0			2	0	0	1,384	ECDSA
DFHSNP	Any	CICS	0	0			2	0	0	13,264	ERDSA
DFHSNSE			0					0	0		None
DFHSTP	Below	CICS	0					0	0		None
DFHSZRMP	Any	CICS	0	0			2	0	0	213,232	ERDSA
DFHTACP	Below	CICS	0	0			2	0	0	5,672	CDSA
DFHTAJP	Below	CICS	0	0			2	0	0	1,736	ECDSA
DFHTBS	Any	CICS	0					0	0		None
DFHTCRP	Below	CICS	0	0			2	0	0	25,776	ERDSA
DFHTDRP	Below	CICS	0	0			2	0	0	6,432	ERDSA
DFHTEP	Any	CICS	0	0			2	0	0	2,592	ECDSA
DFHTEPT	Any	CICS	0	0			2	0	0	3,480	ECDSA
DFHTFP	Any	CICS	0	0			2	0	0	7,744	ECDSA
DFHTOR	Any	CICS	0	0			2	0	0	57,920	ERDSA
DFHTORP	Below	CICS	0	0			2	0	0	560	ERDSA
DFHTPQ	Any	CICS	0					0	0		None
DFHTPR	Any	CICS	0					0	0		None
DFHTPS	Any	CICS	0					0	0		None
DFHTRAP	Any	CICS	0					0	0		None
DFHTSDQ	Any	CICS	0					0	0		None
DFHUCNV	Any	CICS	0					0	0		None
DFHWBA	Any	CICS	0					0	0		None
DFHWBADX	Any	CICS	0					0	0		None
DFHWBAHX	Any	CICS	0					0	0		None

Figure 46. The Programs Report

Table 161. Fields in the Programs Report

Field Heading	Description
Programs	
Program Name	The name of the program. Source field: EXEC CICS INQUIRE PROGRAM
Data Loc	The storage location that the program is able to accept. Source field: EXEC CICS INQUIRE PROGRAM DATALOCATION
Exec Key	The access key in which the program will execute. Source field: EXEC CICS INQUIRE PROGRAM EXECKEY

Table 161. Fields in the Programs Report (continued)

Field Heading	Description
Times Used	The number of times CICS tasks within the system have issued load requests to the loader domain to obtain access to a usable instance of this program. These load requests may cause the loader domain to issue an MVS LOAD. Source field: LDRTU
Times Fetched	The number of times the loader domain has issued an MVS LOAD request to load a copy of the program from the DFHRPL library concatenation into CICS managed storage. Source field: LDRFC
Total Fetch Time	The time taken to perform all fetches for this program. Source field: LDRFT
Average Fetch Time	The average time taken to perform a fetch of the program. Source field: (LDRFT / LDRFC)
RPL Offset	The offset into the DFHRPL DD concatenation of the library from which the program was last loaded or will be loaded when next required (non-LPA resident modules only). **Note:** The offset values begin with zero for the first partitioned data set in the concatenation and thus this field may not be used to deduce whether a copy of the program is available to the loader domain. Source field: LDRRPLO
Times Newcopy	The number of times a NEWCOPY has been requested against this program. Source field: LDRTN
Times Removed	The number of times an instance of this program has been removed from CICS managed storage due to the actions of the Dynamic Program Storage Compression (DPSC) mechanism. Source field: LDRRPC
Program Size	The size of the program in bytes, if known (otherwise zero). Source field: LDRPSIZE
Program Location	The location of the current storage resident instance of the program, if any. It has one of the following values: • None - No current copy • CDSA - Current copy is in the CDSA • SDSA - Current copy is in the SDSA • RDSA - Current copy is in the RDSA • ECDSA - Current copy is in the ECDSA • ESDSA - Current copy is in the ESDSA • ERDSA - Current copy is in the ERDSA • LPA - Current copy is in the LPA • ELPA - Current copy is in the ELPA Source field: LDRLOCN

Program Totals Report

Figure 47 on page 551 shows the format of the Program Totals Report. The field headings and contents are described in Table 162 on page 551.

```
Applid IYK2Z2G1  Sysid JOHN  Jobname CI13JTD5    Date 12/10/1998  Time 09:58:34                CICS 5.3.0                PAGE    26

Program Totals

               Programs. . . . . . . :       1,208
               Assembler . . . . . . :       1,046
               C . . . . . . . . . . :           6
               COBOL . . . . . . . . :          49
               Java (JVM). . . . . . :           4
               LE/370. . . . . . . . :          10
               PL1 . . . . . . . . . :          86
               Remote. . . . . . . . :           0
               Not Deduced . . . . . :           7
               Maps. . . . . . . . . :          69
               Partitionsets . . . . :           1
               ____________________________________
               Total . . . . . . . . :       1,278


               CDSA Programs . . . . :           0
               SDSA Programs . . . . :           0
               RDSA Programs . . . . :           3

               ECDSA Programs. . . . :          11
               ESDSA Programs. . . . :           0
               ERDSA Programs. . . . :          34

               LPA Programs. . . . . :           0
               ELPA Programs . . . . :           0

               Unused Programs . . . :           2
               Not Located Programs. :       1,228
               ____________________________________
               Total . . . . . . . . :       1,278
```

Figure 47. The Program Totals Report

Table 162. Fields in the Program Totals Report

Field Heading	Description
Program Totals	
Programs - Assembler	The current total number of programs defined to CICS as Assembler programs. Source field: EXEC CICS INQUIRE PROGRAM LANGDEDUCED(cvda) RUNTIME(cvda).
Programs - C	The current total number of programs defined to CICS as C programs. Source field: EXEC CICS INQUIRE PROGRAM LANGDEDUCED(cvda) RUNTIME(cvda).
Programs - COBOL	The current total number of programs defined to CICS as COBOL programs. Source field: EXEC CICS INQUIRE PROGRAM LANGDEDUCED(cvda) RUNTIME(cvda).
Programs - Java	The current total number of programs defined to CICS as Java programs. Source field: EXEC CICS INQUIRE PROGRAM LANGDEDUCED(cvda) RUNTIME(cvda).
Programs - LE/370	The current total number of programs defined to CICS as LE/370 programs. Source field: EXEC CICS INQUIRE PROGRAM LANGDEDUCED(cvda) RUNTIME(cvda).
Programs - PL1	The current total number of programs defined to CICS as PL/1 programs. Source field: EXEC CICS INQUIRE PROGRAM LANGDEDUCED(cvda) RUNTIME(cvda).

Table 162. Fields in the Program Totals Report (continued)

Field Heading	Description
Programs - Remote	The current total number of programs defined to CICS as remote programs. Source field: EXEC CICS INQUIRE PROGRAM LANGDEDUCED(cvda) RUNTIME(cvda).
Programs - Not Deduced	The current total number of programs defined to CICS but whose language was not specified in the resource definition. Source field: EXEC CICS INQUIRE PROGRAM LANGDEDUCED(cvda) RUNTIME(cvda).
Maps	The current number of maps defined to CICS.
Partitionsets	The current number of partitionsets defined to CICS.
Total	The total number of programs, maps, and partitionsets defined to CICS.
CDSA Programs	The number of programs, maps, and partitionsets defined to CICS currently residing in the CDSA.
SDSA Programs	The number of programs, maps, and partitionsets defined to CICS currently residing in the SDSA.
RDSA Programs	The number of programs, maps, and partitionsets defined to CICS currently residing in the RDSA.
ECDSA Programs	The number of programs, maps, and partitionsets defined to CICS currently residing in the ECDSA.
ESDSA Programs	The number of programs, maps, and partitionsets defined to CICS currently residing in the ESDSA.
ERDSA Programs	The number of programs, maps, and partitionsets defined to CICS currently residing in the ERDSA.
LPA Programs	The current number of programs, maps, and partitionsets defined to CICS residing in the LPA.
ELPA Programs	The current number of programs, maps, and partitionsets defined to CICS residing in the ELPA.
Unused Programs	The current number of programs, maps, and partitionsets defined to CICS and which have been located in a concatenation of the DFHRPL DD program library but which have not been used by any CICS task.
Not Located Programs	The current number of programs, maps, and partitionsets defined to CICS but which have not been located in any concatenation of the DFHRPL DD program library.
Total	The total number of programs, maps, and partitionsets defined to CICS.

DFHRPL Analysis Report

Figure 48 on page 553 shows the format of the DFHRPL Analysis Report. The field headings and contents are described in Table 163 on page 553.

```
Applid IYK2Z1V3  Sysid CJB3  Jobname CI07CJB3    Date 08/20/1998  Time 15:33:57                CICS 5.3.0                PAGE    45

DFHRPL Analysis

   RPL                                                 Average
  Offset    Programs    Times Used     Fetches      Fetch Time      Newcopies     Removes

     0             1             2           1  00:00:00.02214             0           0
     1             2             6           1  00:00:00.00422             0           0
     2            52           558           4  00:00:00.03283             0           0
     3             3             4           0                             0           0
     4             0             0           0                             0           0
     5             0             0           0                             0           0
     6             9            13           9  00:00:00.02073             0           0
     7             0             0           0                             0           0
     8             1             0           0                             0           0
     9             9             0           0                             0           0
  Totals          77           583          15                             0           0
```

Figure 48. The DFHRPL Analysis Report

Table 163. Fields in the DFHRPL Analysis Report

Field Heading	Description
DFHRPL Analysis	
RPL Offset	The offset into the DFHRPL DD program library concatenation.
Programs	The current number of programs, maps, and partitionsets defined to CICS and located in this concatenation of the DFHRPL DD program library.
Times Used	The number of times CICS tasks within the system have issued load requests to the loader domain to obtain access to a usable instance of this program that have fetched from this concatenation of the DFHRPL DD program library. Source field: LDRTU
Fetches	The number of times programs were fetched from this concatenation of the DFHRPL DD program library. Source field: LDRFC
Average Fetch Time	The average fetch time for programs fetched from this concatenation of the DFHRPL DD program library. Source field: (LDRFT / LDRFC)
Newcopies	The number of times programs were newcopied which have been fetched from this concatenation of the DFHRPL DD program library. Source field: LDRTN
Removes	The number of times programs were removed from CICS managed storage due to the actions of the Dynamic Program Storage Compression (DPSC) mechanism which had been fetched from this concatenation of the DFHRPL DD program library. Source field: LDRRPC

Programs by DSA and LPA Report

Figure 49 on page 554 shows the format of the Programs by DSA and LPA Report. The field headings and contents are described in Table 164 on page 554.

Applid IYK2Z1V3 Sysid CJB3 Jobname CI07CJB3 Date 08/20/1998 Time 15:33:57 CICS 5.3.0 PAGE 50

Program Name	Concurrency Status	Times Used	Times Fetched	Total Fetch Time	Average Fetch Time	RPL Offset	Times Newcopy	Times Removed	Program Size	Program Location
CEEEV003	Quasi Rent	0	0			9	0	0	1,934,336	ERDSA
CEEEV005	Quasi Rent	0	0			9	0	0	13,720	ERDSA
CEEEV010	Quasi Rent	0	0			9	0	0	225,488	ERDSA
CEELCLE	Quasi Rent	0	0			9	0	0	25,400	ERDSA
CEEPLPKA	Quasi Rent	0	0			9	0	0	851,792	ERDSA
CEEQMATH	Quasi Rent	0	0			9	0	0	544,160	ERDSA
DFHAMP	Quasi Rent	16	1	00:00:00.07161	00:00:00.07161	2	0	0	161,464	ERDSA
DFHAPATT	Quasi Rent	0	0			2	0	0	760	ERDSA
DFHCRNP	Quasi Rent	0	0			2	0	0	11,528	ERDSA
DFHCRQ	Quasi Rent	0	0			2	0	0	872	ERDSA
DFHCRR	Quasi Rent	0	0			2	0	0	4,824	ERDSA
DFHCRSP	Quasi Rent	0	0			2	0	0	3,528	ERDSA
DFHDMP	Quasi Rent	42	0			2	0	0	42,552	ERDSA
DFHD2RP	Quasi Rent	0	0			2	0	0	4,544	ERDSA
DFHEDAD	Quasi Rent	2	1	00:00:00.03720	00:00:00.03720	2	0	0	140,744	ERDSA
DFHEDAP	Quasi Rent	2	1	00:00:00.00422	00:00:00.00422	1	0	0	3,208	ERDSA
DFHEITMT	Quasi Rent	4	0			2	0	0	45,416	ERDSA
DFHEITSP	Quasi Rent	4	1	00:00:00.00627	00:00:00.00627	2	0	0	25,456	ERDSA
DFHEMTD	Quasi Rent	4	0			2	0	0	99,896	ERDSA
DFHEMTP	Quasi Rent	4	0			1	0	0	3,304	ERDSA
DFHPUP	Quasi Rent	14	0			2	0	0	20,904	ERDSA
DFHQRY	Quasi Rent	0	0			2	0	0	3,936	ERDSA
DFHSHSY	Quasi Rent	0	0			2	0	0	632	ERDSA
DFHSNP	Quasi Rent	0	0			2	0	0	13,264	ERDSA
DFHSZRMP	Quasi Rent	0	0			2	0	0	213,232	ERDSA
DFHTCRP	Quasi Rent	0	0			2	0	0	25,776	ERDSA
DFHTDRP	Quasi Rent	0	0			2	0	0	6,432	ERDSA
DFHTOR	Quasi Rent	0	0			2	0	0	57,920	ERDSA
DFHTORP	Quasi Rent	0	0			2	0	0	560	ERDSA
DFHWBGB	Quasi Rent	0	0			2	0	0	752	ERDSA
DFHWBIP	Quasi Rent	0	0			2	0	0	2,896	ERDSA
DFHWBST	Quasi Rent	0	0			2	0	0	11,144	ERDSA
DFHWBTC	Quasi Rent	0	0			2	0	0	72,336	ERDSA
DFHZATA	Quasi Rent	0	0			2	0	0	30,360	ERDSA
DFHZATDY	Quasi Rent	0	0			2	0	0	592	ERDSA
DFHZATR	Quasi Rent	0	0			2	0	0	2,656	ERDSA
DFHZATS	Quasi Rent	0	0			2	0	0	13,072	ERDSA
DFHZCGRP	Quasi Rent	0	0			2	0	0	1,064	ERDSA
DFHZCQ	Quasi Rent	0	0			2	0	0	250,160	ERDSA
DFHZCSTP	Quasi Rent	0	0			2	0	0	656	ERDSA
DFHZNAC	Quasi Rent	0	0			2	0	0	42,240	ERDSA
DFHZXRE	Quasi Rent	0	0			2	0	0	3,608	ERDSA
DFHZXST	Quasi Rent	0	0			2	0	0	9,304	ERDSA
DFH0STAT	Quasi Rent	1	0			3	0	0	277,208	ERDSA
DFH0VZXS	Quasi Rent	0	0			2	0	0	8,352	ERDSA
IGZCPAC	Quasi Rent	0	0			9	0	0	112,792	ERDSA
IGZCPCC	Quasi Rent	0	0			8	0	0	11,784	ERDSA
Totals		93	4				0	0		

Figure 49. The Programs by DSA and RPL Report

Table 164. Fields in the Programs by DSA and RPL Report

Field Heading	**Description**
Programs	
Program Name	The name of the program. Source field: EXEC CICS INQUIRE PROGRAM()
Concurrency Status	The Source field: EXEC CICS INQUIRE PROGRAM() CONCURRENCY(cvda)
Times Used	The number of times CICS tasks within the system have issued load requests to the loader domain to obtain access to a usable instance of this program. These load requests may cause the loader domain to issue an MVS LOAD. Source field: LDRTU
Times Fetched	The number of times the loader domain has issued an MVS LOAD request to load a copy of the program from the DFHRPL library concatenation into CICS managed storage. Source field: LDRFC

Table 164. Fields in the Programs by DSA and RPL Report (continued)

Field Heading	Description
Total Fetch Time	The time taken to perform all fetches for this program. Source field: LDRFT
Average Fetch Time	The average time taken to perform a fetch of the program. Source field: (LDRFT / LDRFC)
RPL Offset	The offset into the DFHRPL DD concatenation of the library from which the program was last loaded or is loaded when next required non-LPA resident modules only. **Note:** The offset values begin with zero for the first partitioned data set in the concatenation and thus this field may not be used to deduce whether a copy of the program is available to the loader domain. Source field: LDRRPLO
Times Newcopy	The number of times a NEWCOPY has been requested against this program. Source field: LDRTN
Times Removed	The number of times an instance of this program has been removed from CICS managed storage due to the actions of the Dynamic Program Storage Compression (DPSC) mechanism. Source field: LDRRPC
Program Size	The size of the program in bytes, if known (otherwise zero). Source field: LDRPSIZE
Program Location	The location of the current storage resident instance of the program, if any. It has one of the following values: • None - No current copy • CDSA - Current copy is in the CDSA • SDSA - Current copy is in the SDSA • RDSA - Current copy is in the RDSA • ECDSA - Current copy is in the ECDSA • ESDSA - Current copy is in the ESDSA • ERDSA - Current copy is in the ERDSA • LPA - Current copy is in the LPA • ELPA - Current copy is in the ELPA Source field: LDRLOCN

Temporary Storage Report

Figure 50 on page 556 shows the format of the Temporary Storage Report. This report report is produced using the EXEC CICS COLLECT STATISTICS TSQUEUE command. The statistics data is mapped by the DFHTSGDS DSECT. The field headings and contents are described in Table 165 on page 556.

```
Applid IYK2Z1V3  Sysid CJB3  Jobname CI07CJB3    Date 08/20/1998  Time 15:33:57              CICS 5.3.0              PAGE    51


Temporary Storage

  Put/Putq main storage requests . . . . . :           0
  Get/Getq main storage requests . . . . . :           0
  Peak storage used for TS Main. . . . . . :           0K
  Current storage used for TS Main . . . . :           0K

  Put/Putq auxiliary storage requests. . . :           5
  Get/Getq auxiliary storage requests. . . :           1

  Times temporary storage queue created. . :           5
  Peak temporary storage queues in use . . :           5
  Current temporary storage queues in use. :           5
  Items in longest queue . . . . . :           1

  Control interval size. . . . . :       4,096
  Control intervals in the DFHTEMP dataset :         359
  Peak control intervals used. . . . . :           2
  Current control intervals in use . . . . :           2
  Available bytes per control interval . . :       4,032
  Segments per control interval. . . . . :          63
  Bytes per segment. . . . . :          64
  Writes bigger than control interval size :           0
  Largest record length written. . . . . :         294
  Times auxiliary storage exhausted. . . . :           0
  Number Temporary storage compressions. . :           0

  Temporary storage strings. . . . . :           1
  Peak Temporary storage strings in use. . :           1
  Temporary storage string waits . . . . . :           0
  Peak users waiting on string . . . . . :           0
  Current users waiting on string. . . . . :           0

  Temporary storage buffers. . . . . :           3
  Temporary storage buffer waits . . . . . :           0
  Peak users waiting on buffer . . . . . :           0
  Current users waiting on buffer. . . . . :           0
  Temporary storage buffer reads . . . . . :           0
  Temporary storage buffer writes. . . . . :           2
  Forced buffer writes for recovery. . . . :           2
  Format writes. . . . . :           0

  I/O errors on the DFHTEMP dataset. . . . :           0

  Shared Pools defined . . . . . :           3
  Shared Pools currently connected . . . . :           2
  Shared temporary storage read requests . :           7
  Shared temporary storage write requests. :          15
```

Figure 50. The Temporary Storage Report

Table 165. Fields in the Temporary Storage Report

Field Heading	Description
Temporary Storage	
Put/Putq main storage requests	The number of records that application programs wrote to main temporary storage. Source field: TSGSTA5F
Get/Getq main storage requests	The number of records that application programs obtained from main temporary storage. Source field: TSGNMG
Peak storage used for TS Main	The peak value, expressed in Kbytes, of the amount of virtual storage used for temporary storage records. Source field: (TSGSTA6F / 1024)
Current storage used for TS Main	The current value, expressed in Kbytes, of the amount of virtual storage used for temporary storage records. Source field: (TSGSTA6A / 1024)

Table 165. Fields in the Temporary Storage Report (continued)

Field Heading	Description
Put/Putq auxiliary storage requests	The number of records that application programs wrote to auxiliary temporary storage. Source field: TSGSTA7F
Get/Getq auxiliary storage requests	The number of records that application programs obtained from auxiliary temporary storage. Source field: TSGNAG
Times temporary storage queue created	The number of times that CICS created individual temporary storage queues. Source field: TSGSTA3F
Peak temporary storage queues in use	The peak number of temporary storage queue names in use at any one time. Source field: TSGQNUMH
Current temporary storage queues in use	The current number of temporary storage queue names in use. Source field: TSGQNUM
Items in longest queue	The peak number of items in any one temporary storage queue. Source field: TSGQINH
Control interval size	The size of VSAM's unit of transmission between DASD and main storage, specified in the CONTROLINTERVALSIZE parameter in the VSAM CLUSTER definition for the temporary storage data set. In general, using large CIs permits more data to be transferred at one time, resulting in less system overhead. Source field: TSGCSZ
Control intervals in the DFHTEMP dataset	The number of control intervals (CIs) available for auxiliary storage. This is the total available space on the temporary storage data set expressed as a number of control intervals. This is not the space remaining at termination. Source field: TSGNCI
Peak control intervals in use	The peak number of control intervals (CIs) containing active data. Source field: TSGNCIAH
Current control intervals in use	The current number of control intervals (CIs) containing active data. Source field: TSGNCIA
Available bytes per control interval	The number of bytes available for use in a DFHTEMP data set control interval (CI). Source field: TSGNAVB
Segments per control interval	The number of segments available in a DFHTEMP data set control interval (CI). Source field: TSGSPCI
Bytes per segment	The number of bytes per segment of the DFHTEMP data set. Source field: TSGBPSEG

Table 165. Fields in the Temporary Storage Report (continued)

Field Heading	Description
Writes bigger than control interval size	The number of writes of records whose length was greater than the control interval (CI) size. Source field: TSGSTABF
Largest record length written	The size, expressed in bytes, of the longest record written to the temporary storage data set. Source field: TSGLAR
Times auxiliary storage exhausted	The number of situations where one or more transactions may have been suspended because of a NOSPACE condition, or (using a HANDLE CONDITION NOSPACE command) may have been forced to abend. Source field: TSGSTA8F
Number Temporary Storage compressions	The number of times that the temporary storage buffers were compressed. Source field: TSGSTA9F
Temporary storage strings	The number of temporary storage strings specified in the TS= system initialization parameter or in the overrides. The number of strings allocated may exceed the number requested. Source field: TSGNVCA
Peak Temporary storage strings in use	The peak number of concurrent I/O operations. If this is significantly less than the number specified in the SIT, consider reducing the SIT value to approach this number. Source field: TSGNVCAH
Temporary storage string waits	The number of I/O requests that were queued because no strings were available. This is zero if the number of strings is the same as the number of buffers. If this is a high percentage (over 30%) of the number of I/O requests, consider increasing the number of strings initially allocated. Source field: TSGVWTN
Peak users waiting on string	The peak number of I/O requests that were queued at any one time because all strings were in use. Source field: TSGVUWTH
Current users waiting on string	The current number of I/O requests that are queued because all strings are in use. Source field: TSGVUWT
Temporary storage buffers	The number of temporary storage buffers specified in the TS= system initialization parameter or in the overrides. The number of buffers allocated may exceed the number requested. Source field: TSGNBCA
Temporary storage buffer waits	The number of times a request was queued because all buffers were allocated to other tasks. A buffer wait occurs if the required control interval is already in a locked buffer, and therefore unavailable, even if there are other buffers available. Source field: TSGBWTN

Table 165. Fields in the Temporary Storage Report (continued)

Field Heading	Description
Peak users waiting on buffer	The peak number of requests queued because no buffers were available. Source field: TSGBUWTH
Current users waiting on buffer	The current number of requests queued because no buffers are available. Source field: TSGBUWT
Temporary storage buffer reads	The number of times a CI has to be read from disk. Increasing the buffer allocation decreases this activity. Source field: TSGTRDN
Temporary storage buffer writes	The number of WRITEs to the temporary storage data set. This includes both WRITEs necessitated by recovery requirements (see next item) and WRITEs forced by the buffer being needed to accommodate another CI. Source field: TSGTWTN
Forced buffer writes for recovery	The subset of the total number of WRITEs caused by recovery being specified for queues. This I/O activity is not affected by buffer allocation. Source field: TSGTWTNR
Format writes	The number of times a new control interval (CI) was successfully written at the end of the data set to increase the amount of available space in the temporary storage data set. A formatted write is attempted only if the current number of control intervals (CIs) available in the auxiliary data set have all been used. Source field: TSGTWTNF
I/O errors on the DFHTEMP dataset	The number of input/output errors which occurred on the temporary storage data set. This should normally be zero. If it is not, inspect the CICS and VSAM messages to determine the cause. Source field: TSGSTAAF
Shared Pools defined	The number of unique Shared TS Queue Pools defined either in the TST with DFHTST TYPE=SHARED, or by using TSMODEL. Source field: TSGSHPDF
Shared Pools currently connected	The number of the TS pools that are actually connected to by this CICS region. Source field: TSGSHPCN
Shared temporary storage read requests	The number of TS READQs from the Shared TS Queue pool of Queueids. Source field: TSGSHRDS
Shared temporary storage write requests	The number of TS WRITEQs to the Shared TS Queue pool of Queueids. Source field: TSGSHWTS

Temporary Storage Queues Report

Figure 51 shows the format of the Temporary Storage Queues Report. This report is produced using the EXEC CICS INQUIRE TSQUEUE command. The field headings and contents are described in Table 166.

```
Applid IYK2Z1V3  Sysid CJB3  Jobname CI07CJB3    Date 08/20/1998  Time 15:33:57                CICS 5.3.0                PAGE    52

Temporary Storage Queues

                  Tsqueue   Number of  Min Item  Max Item    Tsqueue           Lastused
  TSQueue Name    Location    Items     Length    Length     Flength   Tranid  Interval    Recoverable
  ____________________________________________________________________________________________________

  DF000009        Auxiliary       1        320       320        320     ECPM   00:00:38      No
  HISD            Auxiliary       1        384       384        384     ECPM   00:00:19      Yes
  HISM            Auxiliary       1        320       320        320     ECPM   00:00:19      Yes
  HISR            Auxiliary       1         64        64         64     ECPM   00:00:38      Yes
  OSC             Auxiliary       1        128       128        128     ECPM   00:00:19      Yes
```

Figure 51. The Temporary Storage Queues Report

Table 166. Fields in the Temporary Storage Queues Report

Field Heading	Description
Temporary Storage Queues	
Tsqueue Name	The name of the temporary storage queue. Source field: EXEC CICS INQUIRE TSQNAME()
Tsqueue Location	Indicates where the temporary storage queue resides. Source field: EXEC CICS INQUIRE TSQNAME() LOCATION(cvda)
Number of Items	The number of items in the temporary storage queue. Source field: EXEC CICS INQUIRE TSQNAME() NUMITEMS()
Min Item Length	The length of the smallest item in the temporary storage queue. Source field: EXEC CICS INQUIRE TSQNAME() MINITEMLEN()
Max Item Length	The length of the largest item in the temporary storage queue. Source field: EXEC CICS INQUIRE TSQNAME() MAXITEMLEN()
Tsqueue Flength	The total length of all the items in the temporary storage queue. Source field: EXEC CICS INQUIRE TSQNAME() FLENGTH()
Tranid	The name of the transaction which created the temporary storage queue. Source field: EXEC CICS INQUIRE TSQNAME() TRANSID()
Lastused Interval	The interval since the temporary storage queue was last referenced. Source field: EXEC CICS INQUIRE TSQNAME() LASTUSEDINT()
Recoverable	Indicates whether the temporary storage queue is recoverable. Source field: EXEC CICS INQUIRE TSQNAME() RECOVSTATUS()

Tsqueue Totals Report

Figure 52 on page 561 shows the format of the Tsqueue totals report. The field headings and contents are described in Table 167 on page 561.

```
Tsqueue Totals

  Current temporary storage queues . . . . . . . . . . . . . . . :          5

  Current auxiliary temporary storage queues . . . . . . . . . . :          5
  Current items in auxiliary temporary storage queues. . . . . . :          5
  Average items per auxiliary temporary storage queue. . . . . . :          1

  Current main temporary storage queues. . . . . . . . . . . . . :          0
  Current items in main temporary storage queues . . . . . . . . :          0
  Average items per main temporary storage queue . . . . . . . . :          0
```

Figure 52. The TSQueue Totals Report

Table 167. Fields in the Tsqueue Totals Report

Field Heading	Description
Tsqueue Totals	
Current temporary storage queues	The total number of temporary storage queues in use.
Current auxiliary temporary storage queues	The total number of temporary storage queues in auxiliary storage.
Current items in auxiliary temporary storage queues	The total number of items in temporary storage queues in auxiliary storage.
Average items per auxiliary temporary storage queue	The average number of items in each temporary storage queue in auxiliary storage.
Current main temporary storage queues	The total number of temporary storage queues in main storage.
Current items in main temporary storage queues	The total number of items in temporary storage queues in main storage.
Average items per main temporary storage queue	The average number of items in each temporary storage queue in main storage.

Temporary Storage Queues by Shared TS Pool

Figure 53 on page 562 shows the format of the Temporary storage queues by shared ts pool report. This report is produced using a combination of the EXEC CICS INQUIRE TSPOOL and EXEC CICS INQUIRE TSQUEUE commands. The field headings and contents are described in Table 168 on page 562.

```
Temporary Storage Queues by Shared TS Pool

  Shared TS Pool Name . . . . . :  TSPOOL2          Connection Status. . . . :  CONNECTED

                      Number of  Min Item  Max Item    Tsqueue
      TSQueue Name      Items     Length    Length     Flength
      ____________________________________________________________

      SHAR1               3         11        16          43
      SHAR3               7         16        16         112
      ____________________________________________________________
                         10

  Shared TS Pool Name . . . . . :  TSPOOL3          Connection Status. . . . :  UNCONNECTED

  Shared TS Pool Name . . . . . :  TSPOOL4          Connection Status. . . . :  CONNECTED

                      Number of  Min Item  Max Item    Tsqueue
      TSQueue Name      Items     Length    Length     Flength
      ____________________________________________________________

      AAAA                6          8        13          53
      SHARB               3         16        16          48
      SHARE               2         16        16          32
      ____________________________________________________________
                         11
```

Figure 53. The Temporary Storage Queues by Shared TS Pool Report

Table 168. Fields in the Tsqueue by Shared TS Pool Report

Field Heading	Description
Temporary Storage Queues by Shared TS Pool	
Shared TS Pool Name	The name of the shared temporary storage pool. Source field: EXEC CICS INQUIRE TSPOOL()
Connection Status	Indicates the connection status of the pool. Source field: EXEC CICS INQUIRE TSPOOL() CONNSTATUS(cvda)
TSQueue Name	The name of the temporary storage queue in this pool. Source field: EXEC CICS INQUIRE TSQNAME()
Number of Items	The number of items in the temporary storage queue. Source field: EXEC CICS INQUIRE TSQNAME() NUMITEMS()
Min Item Length	The length of the smallest item in the temporary storage queue. Source field: EXEC CICS INQUIRE TSQNAME() MINITEMLEN()
Max Item Length	The length of the largest item in the temporary storage queue. Source field: EXEC CICS INQUIRE TSQNAME() MAXITEMLEN()
Tsqueue Flength	The total length of all the items in the temporary storage queue. Source field: EXEC CICS INQUIRE TSQNAME() FLENGTH()

Transient Data Report

Figure 54 shows the format of the Transient Data Report. This report is produced using the EXEC CICS COLLECT STATISTICS TDQUEUE command. The statistics data is mapped by the DFHTQGDS DSECT. The field headings and contents are described in Table 169.

```
Applid IYK2Z1V3  Sysid CJB3  Jobname CI07CJB3    Date 08/20/1998  Time 15:33:57                CICS 5.3.0                PAGE    55


Transient Data

  Transient data reads . . . . . . . . . . :           0
  Transient data writes. . . . . . . . . . :           0
  Transient data formatting writes . . . . :           0

  Control interval size. . . . . . . . . . :       1,536
  Control intervals in the DFHINTRA dataset:       3,900
  Peak control intervals used. . . . . . . :           1
  Times NOSPACE on DFHINTRA occurred . . . :           0

  Transient data strings . . . . . . . . . :           3
  Times Transient data string in use . . . :           0
  Peak Transient data strings in use . . . :           0
  Times string wait occurred . . . . . . . :           0
  Peak users waiting on string . . . . . . :           0

  Transient data buffers . . . . . . . . . :           5
  Times Transient data buffer in use . . . :           0
  Peak Transient data buffers in use . . . :           0
  Peak buffers containing valid data . . . :           0
  Times buffer wait occurred . . . . . . . :           0
  Peak users waiting on buffer . . . . . . :           0

  I/O errors on the DFHINTRA dataset . . . :           0
```

Figure 54. The Transient Data Report

Table 169. Fields in the Transient Data Report

Field Heading	Description
Transient Data	
Transient data reads	The number of times a CI has to be read from disk. Increasing the buffer allocation decreases this activity. Source field: TQGACTGT
Transient data writes	The number of WRITEs to the intrapartition transient data set. This includes both WRITEs needed for recovery (see below) and WRITEs forced by the buffer being needed to accommodate another CI. I/O activity caused by the latter reason can be minimized by increasing the buffer allocation. Source field: TQGACTPT
Transient data formatting writes	The number of times a new CI was written at the end of the data set in order to increase the amount of available space. Source field: TQGACTFT
Control interval size	The size of the control interval, expressed in bytes. Source field: TQGACISZ
Control intervals in the DFHINTRA dataset	The current number of control intervals active within the intrapartition data set, DFHINTRA. Source field: TQGANCIS
Peak control intervals used	The peak value of the number of control intervals concurrently active in the system. Source field: TQGAMXCI

Table 169. Fields in the Transient Data Report (continued)

Field Heading	Description
Times NOSPACE on DFHINTRA occurred	The number of times that a NOSPACE condition has occurred. Source field: TQGANOSP
Transient data strings	The number of strings currently active. Source field: TQGSTSTA
Times Transient data string in use	The number of times a string was accessed. Source field: TQGSTNAL
Peak Transient data strings in use	The peak number of strings concurrently accessed in the system. Source field: TQGSMXAL
Times string wait occurred	The number of times that tasks had to wait because no strings being available. Source field: TQGSTNWT
Peak users waiting on string	The peak number of concurrent string waits in the system. Source field: TQGSMXWT
Transient data buffers	The number of transient data buffers specified in the system initialization table (SIT) or in the SIT overrides. The number of buffers allocated may exceed the number requested. Source field: TQGANBFA
Times Transient data buffer in use	The number of times intrapartition buffers have been accessed. Source field: TQGATNAL
Peak Transient data buffers in use	The peak value of the number of concurrent intrapartition buffer accesses. Source field: TQGAMXAL
Peak buffers containing valid data	The peak number of intrapartition buffers that contain valid data. Source field: TQGAMXIU
Times buffer wait occurred	The number of times a request was queued because all buffers were allocated to other tasks. A buffer wait occurs if the required control interval is already in a locked buffer, and therefore unavailable, even if there are other buffers available. Source field: TQGATNWT
Peak users waiting on buffer	The peak number of requests queued because no buffers were available. Source field: TQGAMXWT
I/O errors on the DFHINTRA dataset	The number of input/output errors that have occurred on the DFHINTRA dataset during this run of CICS. Source field: TQGACTIO

Transient Data Queues Report

Figure 55 shows the format of the Transient Data Queues Report. This report is produced using a combination of the EXEC CICS INQUIRE TDQUEUE and EXEC CIC COLLECT STATISTICS TDQUEUE commands. The statistics data is mapped by the DFHTQRDS DSECT. The field headings and contents are described in Table 170.

```
Applid IYK2Z1V3  Sysid CJB3  Jobname CI07CJB3    Date 08/20/1998  Time 15:33:57                CICS 5.3.0                PAGE    56

Transient Data Queues

  Dest  Queue       Tdqueue     Tdqueue     Tdqueue   Indirect  Remote  Remote  Current  No. of  Trigger  <--------- ATI ---------->
   Id   Type        Writes       Reads      Deletes     Name    System   Name    Items  Triggers  Level   Fcty  Term  Tran   Userid
  ____  __________  ________  ________  ________  ____________  ______  ______  _______  ________  _______  ____  ____  ____  ________

  CADL  Indirect           2         0         0    CSSL
  CAFF  Extra              0         0
  CAIL  Indirect           0         0         0    CSSL
  CCPI  Indirect           0         0         0    CSSL
  CCSE  Indirect           0         0         0    CCSO
  CCSO  Extra              0         0
  CCZM  Indirect           0         0         0    CSSL
  CDBC  Indirect           0         0         0    CSSL
  CDB2  Indirect           0         0         0    CSSL
  CDUL  Indirect           0         0         0    CSSL
  CESE  Extra              0         0
  CESO  Extra              0         0
  CKMQ  Extra              0         0
  CKQQ  Intra              0         0         0                                    0        0        1                      CBAKER
  CMIG  Indirect           0         0         0    CSSL
  CPLD  Indirect           0         0         0    CPLI
  CPLI  Extra              0         0
  CRDI  Indirect           2         0         0    CSSL
  CRPO  Extra              0         0
  CSCC  Indirect           0         0         0    CSSL
  CSCS  Indirect           0         0         0    CSSL
  CSDH  Indirect           0         0         0    CSSL
  CSDL  Indirect           6         0         0    CSSL
  CSFL  Indirect           0         0         0    CSSL
  CSKL  Indirect           0         0         0    CSSL
  CSML  Indirect           0         0         0    CSSL
  CSMT  Indirect           0         0         0    CSSL
  CSNE  Indirect           0         0         0    CSSL
  CSOO  Indirect           0         0         0    CSSL
  CSPL  Indirect           0         0         0    CSSL
  CSQL  Indirect           0         0         0    CSSL
  CSRL  Indirect           0         0         0    CSSL
  CSSH  Indirect           0         0         0    CSSL
  CSSL  Extra             10         0
  CSTL  Indirect           0         0         0    CSSL
  CSZL  Indirect           0         0         0    CSSL
  CSZX  Intra              0         0         0                                    0        0        1              CZUX    CBAKER
  CWBO  Indirect           0         0         0    CSSL
  CXRF  Extra              0         0
  LOGA  Extra              0         0
  L86O  Intra              0         0         0                                    0        0       30  TERM  L86O  AORQ
  L86P  Intra              0         0         0                                    0        0        1  TERM  L86P  TDWT
  REX1  Extra              0         0
  REX2  Extra              0         0
  REX3  Extra              0         0
  REX4  Extra              0         0
  REX5  Extra              0         0
  REX6  Extra              0         0
  TCPM  Extra              0         0

  ____________________  ____      ____      ____
                          20         0         0
```

Figure 55. Transient Data Queues Report

Table 170. The Fields in the Transient Data Queue Report

Field Heading	Description
Transient Data Queues	
Dest Id	The destination identifier (queue). Source field: EXEC CICS INQUIRE TDQUEUE()

Table 170. The Fields in the Transient Data Queue Report (continued)

Field Heading	Description
Queue Type	The queue type, extrapartition, intrapartition, indirect or remote. Source field: EXEC CICS INQUIRE TDQUEUE() TYPE(cvda)
Tdqueue Writes	The number of write requests. Source field: TQRWRITE
Tdqueue Reads	The number of read requests. Source field: TQRREAD
Tdqueue Deletes	The number of delete requests. Source field: TQRDELET
Indirect Name	The name of the indirect queue. Source field: TQRIQID
Remote System	The remote connection name (sysid) of the system for this queue. Source field: TQRRSYS
Remote Name	The remote queue name for this queue. Source field: TQRRQID
Current Items	The current number of items in this intrapartition queue. Source field: TQRCNITM
No.of triggers	The number of times a trigger transaction has been attached. Source field: TQRTRIGN
Trigger Level	The number of items that must be in this queue before automatic transaction initiation (ATI) occurs. Source field: TQRTRIGL
ATI Fcty	Indicates whether this queue has a terminal or session associated with it. Source field: EXEC CICS INQUIRE TDQUEUE() ATIFACILITY(cvda)
ATI Term	The name of the terminal or session associated with this queue. Source field: EXEC CICS INQUIRE TDQUEUE() ATITERMID()
ATI Tran	The name of the transaction to be attached when the trigger level for this queue is reached. Source field: TQRATRAN
ATI Userid	The user identifier associated with this queue. Source field: EXEC CICS INQUIRE TDQUEUE() ATIUSERID()

Transient Data Queue Totals Report

Figure 56 on page 567 shows the format of the Transient Data Queues Totals Report. The field headings and contents are described in Table 171 on page 567.

```
Applid IYK2Z1V3  Sysid CJB3  Jobname CI07CJB3    Date 08/20/1998  Time 15:33:57     CICS 5.3.0      PAGE    58


Tdqueue Totals

                          No. of      Tdqueue      Tdqueue      Tdqueue
  Tdqueue Type          Tdqueues      Writes         Reads      Deletes

  Intrapartition . . . :        4           0            0            0
  Extrapartition . . . :       23          10            0
  Indirect . . . . . . :       28          10            0            0
  Remote . . . . . . . :        0           0            0            0
  Total. . . . . . . . :       55
```

Figure 56. The Transient Data Queue Totals Report

Table 171. Fields in the Transient Data Queue Totals Report

Field Heading	Description
Tdqueue Totals	
Tdqueue Type	The queue type, extrapartition, intrapartition, indirect or remote. Source field: EXEC CICS INQUIRE TDQUEUE() TYPE(cvda)
No. of Tdqueues	The number of queues defined as this type.
Tdqueue Writes	The total number of write requests. Source field: TQRWRITE
Tdqueue Reads	The total number of read requests. Source field: TQRREADS
Tdqueue Deletes	The total number of delete requests. Source field: TQRDELET

Journalnames Report

Figure 57 on page 568 shows the format of the Journalnames Report. This report is produced using a combination of the EXEC CICS INQUIRE JOURNALNAME and EXEC CICS COLLECT STATISTICS JOURNALNAME commands. The statistics data is mapped by the DFHLGRDS DSECT. The field headings and contents are described in Table 172 on page 568.

Journalnames

Journal Name	Journal Status	Journal Type	Logstream Name	Write Requests	Bytes Written	Average Bytes	Buffer Flushes
DFHJ02	Enabled	SMF		17	16,345	961	17
DFHJ04	Enabled	MVS	CBAKER.IYK2Z1V2.DFHJ04	30	29,090	969	30
DFHJ05	Enabled	MVS	CBAKER.IYK2Z1V2.DFHJ05	15	14,545	969	15
DFHJ06	Enabled	MVS	CBAKER.IYK2Z1V2.DFHJ06	15	14,545	969	15
DFHJ08	Enabled	MVS	CBAKER.IYK2Z1V2.DFHJ08	0	0	0	0
DFHJ15	Enabled	MVS	CBAKER.IYK2Z1V2.DFHJ05	45	43,635	969	45
DFHJ16	Enabled	MVS	CBAKER.IYK2Z1V2.DFHJ06	15	14,545	969	15
DFHLOG	Enabled	MVS	CBAKER.IYK2Z1V2.DFHLOG	N/A	N/A	N/A	N/A
DFHSHUNT	Enabled	MVS	CBAKER.IYK2Z1V2.DFHSHUNT	N/A	N/A	N/A	N/A

Figure 57. The Journalnames Report

Table 172. Fields in the Journalnames Report

Field Heading	Description
Journalnames	
Journal Name	The name of the journal. Source field: EXEC CICS INQUIRE JOURNALNAME()
Journal Status	The current journal status. Source field: EXEC CICS INQUIRE JOURNALNAME() STATUS(cvda)
Journal Type	The type of journal, MVS, SMF or Dummy. Source field: EXEC CICS INQUIRE JOURNALNAME() TYPE(cvda)
Logstream Name	The name of the logstream associated with this journal (MVS journals only). Source field: LGRSTREAM
Write Requests	The number of write requests for this journal. Source field: LGRWRITES
Bytes Written	The number of bytes written to this journal. Source field: LGRBYTES
Average Bytes	The average number of written to this journal per request. Source field: (LGRBYTES / LGRWRITES)
Buffer Flushes	The number of buffer flush requests issued for this journal. Source field: LGRBUFLSH

Logstreams Report

Figure 58 on page 569 shows the format of the Logstreams Report. This report is produced using a combination of the EXEC CICS INQUIRE STREAMNAME and EXEC CICS COLLECT STATISTICS STREAMNAME commands. The statistics data is mapped by the DFHLGSDS DSECT. The field headings and contents are described in Table 173 on page 569.

```
Applid IYK2Z1V3  Sysid CJB3  Jobname CI07CJB3    Date 08/20/1998  Time 15:33:57                CICS 5.3.0                PAGE    60

Logstreams - Resource

                                  Use                Sys                        Max Block  DASD   Retention   Auto   Stream   Browse   Browse
Logstream Name                   Count  Status      Log  Structure Name          Length    Only    Period     Delete Deletes  Starts   Reads
CBAKER.IYK2Z1V2.DFHJ04               1  OK          NO                           32,000   YES          21   YES      N/A      N/A      N/A
CBAKER.IYK2Z1V2.DFHJ05               2  OK          NO                           48,000   YES          14   YES      N/A      N/A      N/A
CBAKER.IYK2Z1V2.DFHJ06               2  OK          NO   LOG_GENERAL_001         64,000   NO            0   NO       N/A      N/A      N/A
CBAKER.IYK2Z1V2.DFHJ08               1  OK          NO   LOG_GENERAL_001         64,000   NO            0   NO       N/A      N/A      N/A
CBAKER.IYK2Z1V2.DFHLOG               1  OK          YES  LOG_GENERAL_005         64,000   NO            0   NO         0       46        0
CBAKER.IYK2Z1V2.DFHSHUNT             1  OK          YES  LOG_GENERAL_006         64,000   NO            0   NO         0        0        0
```

Figure 58. The Logstreams Report

Table 173. Fields in the Logstreams Report

Field Heading	Description
Logstreams - Resource	
Logstream Name	The name of the logstream. Source field: EXEC CICS INQUIRE STREAMNAME()
Use Count	The current use count of the logstream. Source field: EXEC CICS INQUIRE STREAMNAME() USECOUNT()
Status	The current status of the logstream. Source field: EXEC CICS INQUIRE STREAMNAME() STATUS()
Sys Log	Indicates if the log stream forms part of the System Log. Source field: LGSSYSLG
Structure Name	The coupling facility (CF) structure name for the log stream. The structure name is only applicable to coupling facility type logstreams. Source field: LGSSTRUC
Max Block Length	The maximum block size allowed by the MVS Logger for the log stream. Source field: LGSMAXBL
DASD Only	Indicates the type of log stream. If set to 'YES' the log stream is of type DASDONLY. If set to 'NO' the log stream is of type coupling facility (CF). Source field: LGSDONLY
Retention Period	The log stream retention period (in days) that the data must be kept before it can be physically deleted by the MVS Logger. Source field: LGSRETPD
Auto Delete	The log data auto delete indicator. If set to 'YES' the MVS Logger automatically deletes the data as it matures beyond the retention period, irrespective of any logstream delete calls. If set to 'NO' the data is only deleted when a logstream delete call is issued and the data has matured beyond the retention period. Source field: LGSAUTOD
Stream Deletes	The number of delete (IXGDELET) requests issued for this logstream. Source field: LGSDELETES
Browse Starts	The number of browse start requests issued for this logstream. Source field: LGSBRWSTRT

Table 173. Fields in the Logstreams Report (continued)

Field Heading	Description
Browse Reads	The number of browse read requests issued for this logstream. Source field: LGSBRWREAD

Figure 59 shows the format of the Logstreams Report. This report is produced using a combination of the EXEC CICS INQUIRE STREAMNAME and EXEC CICS COLLECT STATISTICS STREAMNAME commands. The statistics data is mapped by the DFHLGSDS DSECT. The field headings and contents are described in Table 174.

```
Applid IYK2Z1V3  Sysid CJB3  Jobname CI07CJB3    Date 11/23/1998  Time 10:47:07                CICS 5.3.0             PAGE     4

Logstreams - Requests

                                  Write                          Average     Buffer      Buffer     Force   Current    Peak   Retry
Logstream Name                  Requests    Bytes Written        Bytes      Appends   Full Waits   Waits   Waiters  Waiters  Errors
_____________________________________________________________________________________________________________________________

CBAKER.IYK2Z1V3.DFHJ04               30           30,366        1,012          30            0       0         0        0       0
CBAKER.IYK2Z1V3.DFHJ05               60           60,656        1,010          60            0       0         0        0       1
CBAKER.IYK2Z1V3.DFHJ06               30           30,290        1,009          30            0       0         0        0       0
CBAKER.IYK2Z1V3.DFHJ08                0                0            0           0            0       0         0        0       0
CBAKER.IYK2Z1V3.DFHLOG               10            6,578          657          20            0       0         0        0       0
CBAKER.IYK2Z1V3.DFHSHUNT              0                0            0           0            0       0         0        0       0
```

Figure 59. The Logstreams Report

Table 174. Fields in the Logstreams Report

Field Heading	Description
Logstreams - Requests	
Logstream Name	The name of the logstream. Source field: EXEC CICS INQUIRE STREAMNAME()
Write Requests	The number of IXGWRITE requests issued to this logstream. Source field: LGSWRITES
Bytes Written	The number of bytes written to this logstream. Source field: LGSBYTES
Average Bytes	The average number of bytes written to this logstream per request. Source field: (LGSBYTES / LGSWRITES)
Buffer Appends	The number of occasions on which a journal record was successfully appended to the current log stream buffer. Source field: LGSBUFAPP
Buffer Full Waits	The number of times buffer full has occurred for this logstream. Source field: LGSBUFWAIT
Force Waits	The total number of force waits for this logstream. Source field: LGSTFCWAIT
Peak Waiters	The peak number of force waiters for this logstream. Source field: LGSPKFWTRS

Table 174. Fields in the Logstreams Report (continued)

Field Heading	Description
Retry Errors	The number of occasions on which MVS system logger retryable errors occurred when a block of data was being written to the log stream. Source field: LGSRTYERRS

Autoinstall and VTAM Report

Figure 60 shows the format of the Autoinstall and VTAM Report. This report is produced using a combination of the EXEC CICS INQUIRE AUTOINSTALL, INQUIRE SYSTEM, INQUIRE VTAM, and the EXEC CICS COLLECT STATISTICS AUTOINSTALL, PROGAUTO and VTAM commands. The statistics data is mapped by the DFHA03DS, DFHA04DS and DFHPGGDS DSECTs. The field headings and contents are described in Table 175 on page 572.

```
Applid IYK2Z1V3  Sysid CJB3  Jobname CI07CJB3    Date 08/20/1998  Time 15:33:57                CICS 5.3.0                PAGE    62


Program Autoinstall

  Program Autoinstall Status. . . . :  INACTIVE
  Autoinstall Program . . . . . . . :  DFHPGADX
  Catalog Program Definitions . . . :  MODIFY

  Autoinstalls attempted. . . . . . :           0
  Autoinstalls rejected . . . . . . :           0
  Autoinstalls failed . . . . . . . :           0


Terminal Autoinstall

  Terminal Autoinstall Status . . . :  ENABLED
  Autoinstall Program . . . . . . . :  DFHZATDY
  Current Autoinstall Requests. . . :           0

  Autoinstalls Attempted. . . . . . :           0
  Autoinstalls Rejected . . . . . . :           0
  Autoinstalls Deleted. . . . . . . :           0

  Peak Concurrent Autoinstalls. . . :           0
  Times Peak Concurrent reached . . :           1
  Times SETLOGON HOLD issued. . . . :           0

  Number of Queued Logons . . . . . :           0
  Peak Number of Queued Logons. . . :           0
  Times Peak Queued Logons reached. :           1


VTAM

  VTAM Open Status. . . . . . . . . :  OPEN
  Dynamic open count. . . . . . . . :           0
  VTAM Short-on-Storage . . . . . . :           0

  MAX RPLs            . . . . . . . :           1
  Times at MAX RPLs . . . . . . . . :          17

  Current LUs in session. . . . . . :           2
  Peak LUs in session . . . . . . . :           2

  Generic Resource name . . . . . . :
  Generic Resource status . . . . . :

  Persistent Session Inquire count. :           0
  Persistent Session NIB count. . . :           0
  Persistent Session Opndst count . :           0
  Persistent Session Unbind count . :           0
  Persistent Session Error count. . :           0
```

Figure 60. The Autoinstall and VTAM Report

Table 175. Fields in the Autoinstall and VTAM Report

Field Heading	Description
Program Autoinstall	
Program Autoinstall Status	Indicates the current status of program autoinstall. Source field: EXEC CICS INQUIRE SYSTEM PROGAUTOINST(cvda)
Autoinstall Program	The name of the user replaceable program autoinstall model definition program. Source field: EXEC CICS INQUIRE SYSTEM PROGAUTOEXIT()
Catalog Program Definitions	Indicates whether autoinstalled program definitions are to be cataloged. Source field: EXEC CICS INQUIRE SYSTEM PROGAUTOCTLG(cvda)
Autoinstalls attempted	The number of program autoinstalls attempted. Source field: PGGATT
Autoinstalls rejected	The number of program autoinstalls rejected by the program autoinstall user replaceable program. Source field: PGGREJ
Autoinstalls failed	The number of program autoinstalls failed for reasons other than being rejected by the program autoinstall user replaceable program. Source field: PGGFAIL
Terminal Autoinstall	
Terminal Autoinstall Status	Indicates the current status of terminal autoinstall. Source field: EXEC CICS INQUIRE AUTOINSTALL ENABLESTATUS(cvda)
Autoinstall Program	The name of the user replaceable terminal autoinstall model definition program. Source field: EXEC CICS INQUIRE AUTOINSTALL PROGRAM()
Current Autoinstall Requests	The number of autoinstall requests currently being processed. Source field: EXEC CICS INQUIRE AUTOINSTALL CURREQS()
Peak Autoinstall Requests	The maximum number of autoinstall requests that can be processed concurrently. Source field: EXEC CICS INQUIRE AUTOINSTALL MAXREQS()
Autoinstalls Attempted	The number of terminal autoinstalls attempted. Source field: A04VADAT
Autoinstalls Rejected	The number of terminal autoinstalls rejected. Source field: A04VADRJ
Autoinstalls Deleted	The number of autoinstalled terminals deleted. Source field: A04VADLO
Peak Concurrent Autoinstalls	The peak number of autoinstall requests processed concurrently. Source field: A04VADPK
Times Peak Concurrent reached	The number of times the peak autoinstall requests was reached. Source field: A04VADPX

Table 175. Fields in the Autoinstall and VTAM Report (continued)

Field Heading	Description
Times SETLOGON HOLD issued	The number of times the VTAM SETLOGON HOLD command was issued. Source field: A04VADSH
Number of Queued Logons	The number of autoinstall attempts that were queued for logon due to the delete being in progress for the same terminal. Source field: A04VADQT
Peak Number of Queued Logons	The peak number of autoinstall attempts that were queued for logon. Source field: A04VADQK
Times Peak Queued Logons reached	The number of time the peak number of autoinstall attempts that were queued for logon was reached, Source field: A04VADQX
VTAM	
VTAM open status	The current status of the connection between CICS and VTAM. Source field: EXEC CICS INQUIRE VTAM OPENSTATUS(cvda)
Dynamic open count	The number of times the VTAM ACB was dynamically opened. Source field: A03DOC
VTAM Short-in-Storage	The number of times that VTAM indicated that there was a temporary VTAM storage problem. Source field: A03VTSOS
MAX RPLs	The maximum number of receive-any request parameter lists (RPLs) that were posted by VTAM on any one dispatch of CICS terminal control. Source field: A03RPLX
Times at MAX RPLs	The number of times the maximum number of receive-any request parameter lists (RPLs) was reached. Source field: A03RPLXT
Current LUs in session	The current number of LUs in session. Source field: A03LUNUM
Peak LUs in session	The peak number of LUs in session. Source field: A03LUHWM
Generic Resource name	The name of the generic resource group which this CICS region requested registration to VTAM. Source field: EXEC CICS INQUIRE VTAM GRNAME()
Generic Resource status	Indicates the status of generic resource registration. Source field: EXEC CICS INQUIRE VTAM GRSTATUS(cvda)
Persistent Session Inquire count	The number of times CICS issued VTAM INQUIRE OPTCD=PERSESS. Source field: A03PSIC
Persistent Session NIB count	The number of VTAM sessions that persisted. Source field: A03PSNC

Table 175. Fields in the Autoinstall and VTAM Report (continued)

Field Heading	Description
Persistent Session Opndst count	The number of persisting sessions that were successfully restored. Source field: A03PSOC
Persistent Session Unbind count	The number of persisting sessions that were terminated. Source field: A03PSUC
Persistent Session Error count	The number of persisting sessions that were already unbound when CICS tried to restore them. Source field: A03PSEC

Connections and Modenames Report

Figure 61 and Figure 62 on page 575 show the format of the Connections and Modenames Report. This report is produced using a combination of the EXEC CICS INQUIRE CONNECTION, EXEC CICS INQUIRE MODENAME and EXEC CICS COLLECT STATISTICS CONNECTION commands. The statistis data is mapped by the DFHA14DS DSECT. The field headings and contents are described in Table 176 on page 575.

```
Applid IYK2Z1V3  Sysid CJB3  Jobname CI07CJB3    Date 08/20/1998  Time 15:33:57                CICS 5.3.0                PAGE    63


Connections

 Connection Name/Netname . . . . . :  CJB2/IYK2Z1V2              Access Method/Protocol. . . . . :    XM

                                                                  Autoinstalled Connection Create Time. . :  00:00:00.00000
 Peak Contention Losers. . . . . :           1
 ATIs satisfied by Losers. . . . . :         0                    Receive Session Count. . . . . :          5
 Peak Contention Winners . . . . . :         1                    Send Session Count . . . . . :           12
 ATIs satisfied by Winners . . . . :         1
 Current AIDs in chain . . . . . :           0                    Generic AIDs in chain. . . . . :          0
 Total number of Bids sent . . . . :         0
 Current Bids in progress. . . . . :         0                    Peak Bids in progress. . . . . :          0

 Total Allocates . . . . . :                 6                    Allocate Max Queue Time. . . . . :        0
 Allocates Queued. . . . . :                 0                    Allocate Queue Limit . . . . . :          0
 Peak Allocates Queued . . . . . :           1
 Allocates Failed - Link . . . . . :         0                    Allocates Rejected - Queue Limit . . :    0
 Allocates Failed - Other. . . . . :         0                    Max Queue Time - Allocate Purge. . . :    0
                                                                  Allocates Purged - Max Queue Time. . :    0
 Transaction Routing Requests. . . :         0
 Function Shipping Requests - FC . :         0                    Allocates Rejected - XZIQUE. . . . . :    0
 Function Shipping Requests - IC . :         0                    XZIQUE - Allocate Purge. . . . . :        0
 Function Shipping Requests - TD . :         0                    Allocates Purged - XZIQUE. . . . . :      0
 Function Shipping Requests - TS . :         0
 Function Shipping Requests - PC . :         0
```

Figure 61. The Connections and Modenames Report

```
Applid IYK2Z1V3  Sysid CJB3  Jobname CI07CJB3    Date 08/20/1998  Time 15:33:57                CICS 5.3.0                PAGE    66

Connections Connection Name/Netname . . . . . :  CJB3/IYK2Z1V3              Access Method/Protocol. . . . . :  VTAM/APPC

                                                                    Autoinstalled Connection Create Time. . :  00:00:00.00000
 Peak Contention Losers. . . . . :           0
 ATIs satisfied by Losers. . . . . :         0                      Receive Session Count . . . . . :          0
 Peak Contention Winners . . . . . :         0                      Send Session Count. . . . . . . . :        0
 ATIs satisfied by Winners . . . . :         0
 Current AIDs in chain . . . . . :           0                      Generic AIDs in chain . . . . . :          0
 Total number of Bids sent . . . . :         0
 Current Bids in progress. . . . . :         0                      Peak Bids in progress . . . . . :          0

 Total Allocates . . . . . :                 0                      Allocate Max Queue Time . . . . . :        0
 Allocates Queued. . . . . :                 0                      Allocate Queue Limit. . . . . . . :        0
 Peak Allocates Queued . . . . . :           0
 Allocates Failed - Link . . . . . :         0                      Allocates Rejected - Queue Limit. :        0
 Allocates Failed - Other. . . . . :         0                      Max Queue Time - Allocate Purge . :        0
                                                                    Allocates Purged - Max Queue Time :        0
 Transaction Routing Requests. . . :         0
 Function Shipping Requests - FC . :         0                      Allocates Rejected - XZIQUE . . . :        0
 Function Shipping Requests - IC . :         0                      XZIQUE - Allocate Purge . . . . . :        0
 Function Shipping Requests - TD . :         0                      Allocates Purged - XZIQUE . . . . :        0
 Function Shipping Requests - TS . :         0
 Function Shipping Requests - PC . :         0


   Modenames

     Modename Connection Name. . . . . :  CJB3
     Modename. . . . . :  SNASVCMG
     Active Sessions . . . . . :          0
     Available Sessions. . . . . :        0
     Maximum Sessions. . . . . :          2
     Maximum Contention Winners. . . . :  1

     Modename Connection Name. . . . . :  CJB3
     Modename. . . . . :
     Active Sessions . . . . . :          0
     Available Sessions. . . . . :        0
     Maximum Sessions. . . . . :          5
     Maximum Contention Winners. . . . :  3
```

Figure 62. The Connections and Modenames Report

Table 176. Fields in the Connections and Modenames Report

Field Heading	Description
Connections	
Connection Name/Netname	The connection name (sysid) and the network name (applid) for the connection. Source field: EXEC CICS INQUIRE CONNECTION() NETNAME()
Access Method/Protocol	The communication access method and protocol used for the connection. Source field: EXEC CICS INQUIRE CONNECTION() ACCESSMETHOD(cvda) PROTOCOL(cvda)
Autoinstalled Connection Create Time	The local time at which this connection was autoinstalled. This field applies to APPC connections only. Source field: A14AICT
Peak Contention Losers	The peak number of contention loser sessions that were in use. Source field: A14E1HWM
ATIs satisfied by Losers	The number of queued allocate requests that have been satisfied by contention loser sessions. Source field: A14ES1
Receive Session Count	The number of receive sessions for this connection. (MRO and LU6.1 connections only) Source field: EXEC CICS INQUIRE CONNECTION() RECEIVECOUNT()

Table 176. Fields in the Connections and Modenames Report (continued)

Field Heading	Description
Peak Contention Winners	The peak number of contention winner sessions that were in use. Source field: A14E2HWM
Send Session Count	The number of send sessions for this connection. (MRO and LU6.1 connections only) Source field: EXEC CICS INQUIRE CONNECTION() SENDCOUNT()
ATIs satisfied by Winners	The number of queued allocate requests that have been satisfied by contention winner sessions. Source field: A14ES2
Current AIDs in chain	The current number of automatic initiate descriptors (AIDs) in the AID chain. Source field: A14EALL
Generic AIDs in chain	The current number of automatic initiate descriptors (AIDs) that are waiting for a session to become available to satisfy the allocate request. Source field: A14ESALL
Total number of Bids sent	The total number of bids sent. Source field: A14ESBID
Current Bids in progress	The current number of bids in progress. Source field: A14EBID
Peak Bids in progress	The peak number of bids that were in progress. Source field: A14EBHWM
Total Allocates	The total number of allocates for this connection. Source field: A14ESTAS
Allocate Max Queue Time	The MAXQTIME value specified for this connection. Source field: A14EMXQT
Allocates Queued	The current number of allocate requests queued for this connection. Source field: A14ESTAQ
Allocates Queue Limit	The QUEUELIMIT value specified for this connection. Source field: A14EALIM
Peak Allocates Queued	The peak number of allocate requests queued for this connection. Source field: A14ESTAM
Allocates Failed - Link	The number of allocate requests that failed due to the connection being released, out of service, or with a closed mode group. Source field: A14ESTAF
Allocates Rejected - Queue Limit	The number of allocate requests that were rejected due to the QUEUELIMIT value being reached. Source field: A14EALRJ

Table 176. Fields in the Connections and Modenames Report (continued)

Field Heading	Description
Allocates Failed - Other	The number of allocate requests that failed due to a session not being currently available for use. Source field: A14ESTAO
Max Queue Time - Allocate Purge	The number of times the allocate request queue has been purged due to the MAXQTIME value being reached. Source field: A14EQPCT
Allocates Purged - Max Queue Time	The total number of allocate requests purged due to the queueing time exceeding the MAXQTIME value. Source field: A14EMQPC
Transaction Routing Requests	The number of transaction routing requests sent across the connection. Source field: A14ESTTC
Function Shipping Requests - FC	The number of file control requests function shipped across the connection. Source field: A14ESTFC
Allocates Rejected - XZIQUE	The number of allocate requests that were rejected by a XZIQUE global user exit. Source field: A14EZQRJ
Function Shipping Requests - IC	The number of interval control requests function shipped across the connection. Source field: A14ESTIC
ZXIQUE - Allocate Purge	The number of times the allocate request queue has been purged by a XZIQUE global user exit. Source field: A14EZQPU
Function Shipping Requests - TD	The number of transient data requests function shipped across the connection. Source field: A14ESTTD
Allocates Purged - ZXIQUE	The total number of allocate requests purged due to a XZIQUE global user exit requesting that the queued allocate requests should be purged. Source field: A14EZQPC
Function Shipping Requests - TS	The number of temporary storage requests function shipped across the connection. Source field: A14ESTTS
Function Shipping Requests - PC	The number of program control requests function shipped across the connection. Source field: A14ESTPC
Modenames	
Modename Connection Name	The name of the connection that owns this mode group entry. Source field: EXEC CICS INQUIRE MODENAME() CONNECTION()
Modename Name	The mode group name. Source field: EXEC CICS INQUIRE MODENAME()

Table 176. Fields in the Connections and Modenames Report (continued)

Field Heading	Description
Active Sessions	The number of sessions in this mode group currently in use. Source field: EXEC CICS INQUIRE MODENAME() ACTIVE()
Available Sessions	The current number of sessions in this mode group (bound). Source field: EXEC CICS INQUIRE MODENAME() AVAILABLE()
Maximum Sessions	The maximum number of sessions defined in this mode group. Source field: EXEC CICS INQUIRE MODENAME() MAXIMUM()
Maximum Contention Winners	The maximum number of sessions in this mode group that are defined to be contention winners. Source field: EXEC CICS INQUIRE MODENAME() MAXWINNERS()

TCP/IP Services Report

Figure 63 and Figure 64 on page 580 show the formats of the TCP/IP Services reports. These reports are produced using a combination of EXEC CICS INQUIRE TCPIPSERVICE and EXEC CICS COLLECT STATISTICS TCPIPSERVICE commands, the statistics data is mapped by the DFHSORDS DSECT. The field headings and contents are described in Table 177 and Table 178 on page 580.

```
TCP/IP Services

Applid IYK2Z1V3  Sysid CJB3  Jobname CI07CJB3    Date 02/18/1999  Time 08:32:40               CICS 5.3.0               PAGE     2


TCP/IP Services
_______________

  TCP/IP     Port                    SSL                  Tsqueue            Service        Open       Open
  Service    Number  IP Address      Type   Backlog  Tran  Prefix   URM       Status         Date       Time
  ____________________________________________________________________________________________________________

  CHRIS1     5060    9.20.2.52       NONE        15  CWXN           DFHWBADX  OPEN        02/18/1999  07:39:04
  CHRIS2     5061    9.20.2.52       NONE         8  CWXN           DFHWBADX  OPEN        02/18/1999  07:39:04
  CHRIS3     5063    9.20.2.52       NONE         8  CWXN           DFHWBADX  OPEN        02/18/1999  07:39:04
  FREDCLIA   5067                    NONE        25  AUTH                     CLOSED
  SAMPLE     5069                    NONE         1  TEST  SAMPLE   12345678  CLOSED
```

Figure 63. The TCP/IP Services Report

Table 177. Fields in the TCP/IP Services Report

Field Heading	Description
TCP/IP Services	
TCP/IP Service	The name of the TCP/IP service. Source field: EXEC CICS INQUIRE TCPIPSERVICE()
Port Number	The number of the port on which CICS is listening on behalf of this service. Source field: EXEC CICS INQUIRE TCPIPSERVICE() PORT()

Table 177. Fields in the TCP/IP Services Report (continued)

Field Heading	Description
IP Address	is the TCP/IP address defined for the TCP/IP stack used for this TCP/IP service. Source field: SOR-IP-ADDRESS
SSL Type	Indicates the level of secure sockets being used for the service. Source field: EXEC CICS INQUIRE TCPIPSERVICE() SSLTYPE(cvda)
Backlog	 Source field: EXEC CICS INQUIRE TCPIPSERVICE() BACKLOG()
Tran	The name of the transaction to be started to process a new request. Source field: EXEC CICS INQUIRE TCPIPSERVICE() TRANSID()
Tsqueue Prefix	 Source field: EXEC CICS INQUIRE TCPIPSERVICE() TSQPREFIX
URM	The name of the service user replaceable module (URM) to be invoked by the attached task. Source field: EXEC CICS INQUIRE TCPIPSERVICE() TSQPREFIX
Service Status	Indicates the current status of this TCP/IP service Source field: EXEC CICS INQUIRE TCPIPSERVICE() OPENSTATUS(cvda)
Open Date	is the date on which this TCP/IP service was opened. If this field is not set, SOR_OPEN_LOCAL contains the hexadecimal value X'0000000000000000', shown in the report as "CLOSED". If the field is set, it contains a date expressed in *mm/dd/yyyy* format. This field contains a valid date if: • The TCP/IP service was open at the time the statistics were taken. • This is an unsolicited statistics request due to the TCP/IP service being closed. Source field: SOR_OPEN_LOCAL
Open Time	is the time at which this TCP/IP service was opened. If this field is not set, SOR_OPEN_LOCAL contains the hexadecimal value X'0000000000000000', shown in the report as "CLOSED". If the field is set, it contains a time expressed as a store clock (STCK) value in local time. This field contains a valid time if: • The TCP/IP service was open at the time the statistics were taken. • This is an unsolicited statistics request due to the TCP/IP service being closed. Source field: SOR_OPEN_LOCAL

```
TCP/IP Services - Requests  TCP/IP
Applid IYK2ZIV3  Sysid CJB3  Jobname CI07CJB3    Date 02/18/1999  Time 08:32:40              CICS 5.3.0              PAGE     3

 TCP/IP Services - Requests
```

TCP/IP Service	Port Number	IP Address	<- Connections -> Current	Peak	Attached	Send Requests	Avg Bytes / Send	Receive Requests	Avg Bytes / Receive
CHRIS1	5060	9.20.2.52	0	2	15	46	1,839	25	324
CHRIS2	5061	9.20.2.52	0	1	1	4	51	1	251
CHRIS3	5063	9.20.2.52	0	1	1	6	842	3	251
FREDCLIA	5067		0	0	0	0	0	0	0
SAMPLE	5069		0	0	0	0	0	0	0
Totals					17	56		29	

Figure 64. The TCP/IP Services Requests Report

Table 178. Fields in the TCP/IP Services Requests Report

Field Heading	Description
TCP/IP Services - Requests	
TCP/IP Service	The name of the TCP/IP service. Source field: EXEC CICS INQUIRE TCPIPSERVICE()
Port Number	The number of the port on which CICS is listening on behalf of this service. Source field: EXEC CICS INQUIRE TCPIPSERVICE() PORT()
IP Address	is the TCP/IP address defined for the TCP/IP stack used for this TCP/IP service. Source field: SOR-IP-ADDRESS
Connections - Current	 Source field: SOR-CURRENT-CONS
Connections - Peak	 Source field: SOR-PEAK-CONS
Attached	 Source field: SOR-TRANS-ATTACHED
Send Requests	 Source field: SOR-SENDS
Avg Bytes / Send	 Source field: (SOR-BYTES-SENT / SOR-SENDS)
Receive Requests	 Source field: SOR-RECEIVES
Avg Bytes / Receive	 Source field: (SOR-BYTES-RECEIVED / SOR-RECEIVES)

LSR Pools Report

Figure 65 shows the format of the LSR pools Report. This report is produced using the EXEC CICS COLLECT STATISTICS LSRPOOL command. The statistics data is mapped by the DFHA08DS DSECT. The field headings and contents are described in Table 179.

```
Applid IYK2Z1V3  Sysid CJB3  Jobname CI07CJB3    Date 08/20/1998  Time 15:33:57                CICS 5.3.0                PAGE    70

LSR Pools

  Pool Number :   2     Time Created :   08:35:07.02513

     Maximum key length . . . . . . . :          16
     Total number of strings  . . . . :           5
     Peak concurrently active strings :           1
     Total requests waited for string :           0
     Peak requests waited for string. :           0

  Buffer Totals

     Data Buffers . . . . . . . . . . :          44                Index Buffers. . . . . . . . . . :          44
     Hiperspace Data Buffers. . . . . :           0                Hiperspace Index Buffers . . . . :           0
       Successful look asides . . . . :         652                  Successful look asides . . . . :       1,360
       Buffer reads . . . . . . . . . :          24                  Buffer reads . . . . . . . . . :           4
       User initiated writes. . . . . :         655                  User initiated writes. . . . . :          31
       Non-user initiated writes. . . :           0                  Non-user initiated writes. . . :           0
       Successful Hiperspace CREADS . :           0                  Successful Hiperspace CREADS . :           0
       Successful Hiperspace CWRITES. :           0                  Successful Hiperspace CWRITES. :           0
       Failing Hiperspace CREADS. . . :           0                  Failing Hiperspace CREADS. . . :           0
       Failing Hiperspace CWRITES . . :           0                  Failing Hiperspace CWRITES . . :           0

  Data Buffer Statistics

                    Hiperspace      Look                    User      Non-User  Successful  Successful     Failing     Failing
    Size  Buffers     Buffers     Asides       Reads      Writes        Writes       CREADS      CWRITES      CREADS      CWRITES

     512        4           0         0           0           0             0           0            0           0            0
    1024        4           0         0           0           0             0           0            0           0            0
    2048        4           0         0           0           0             0           0            0           0            0
    4096        4           0       652          24         655             0           0            0           0            0
    8192        4           0         0           0           0             0           0            0           0            0
   12288        4           0         0           0           0             0           0            0           0            0
   16384        4           0         0           0           0             0           0            0           0            0
   20480        4           0         0           0           0             0           0            0           0            0
   24576        4           0         0           0           0             0           0            0           0            0
   28672        4           0         0           0           0             0           0            0           0            0
   32768        4           0         0           0           0             0           0            0           0            0

  Index Buffer Statistics

                    Hiperspace      Look                    User      Non-User  Successful  Successful     Failing     Failing
    Size  Buffers     Buffers     Asides       Reads      Writes        Writes       CREADS      CWRITES      CREADS      CWRITES

     512        4           0     1,360           4          31             0           0            0           0            0
    1024        4           0         0           0           0             0           0            0           0            0
    2048        4           0         0           0           0             0           0            0           0            0
    4096        4           0         0           0           0             0           0            0           0            0
    8192        4           0         0           0           0             0           0            0           0            0
   12288        4           0         0           0           0             0           0            0           0            0
   16384        4           0         0           0           0             0           0            0           0            0
   20480        4           0         0           0           0             0           0            0           0            0
   24576        4           0         0           0           0             0           0            0           0            0
   28672        4           0         0           0           0             0           0            0           0            0
   32768        4           0         0           0           0             0           0            0           0            0
```

Figure 65. The LSR Pools Report

Table 179. Fields in the LSR Pools Report

Field Heading	Description
LSR Pools	
Pool Number	The identifying number of the LSR pool. This value may be in the range 1 through 8.
Time Created	The time when this LSR pool was created. Source field: A08LBKCD

Table 179. Fields in the LSR Pools Report (continued)

Field Heading	Description
Maximum key length	The length of the largest key of a VSAM data set which may use this LSR pool. Source field: A08BKKYL
Total number of strings	The total number of VSAM strings defined for this LSR pool. Source field: A08BKSTN
Peak concurrently active strings	The maximum number of strings that were active during CICS execution. If you have coded a value for the number of strings the pool is to use, this statistic is always less than or equal to the value you have coded. If your coded value for string numbers is consistently lower than this value in the statistics, you could consider reducing it so that your pool of VSAM strings is not bigger than you need. Source field: A08BKHAS
Total requests waited for strings	The number of requests that were queued because all the strings in the pool were in use. This number reflects the number of requests that were delayed during CICS execution due to a restriction in LSR pool string resources. Source field: A08BKTSW
Peak requests waited for strings	The highest number of requests that were queued at one time because all the strings in the pool were in use. Source field: A08BKHSW
Buffer Totals	
Data Buffers	The number of data buffers specified for the LSR pool. Source field: A08TDBFN
Hiperspace Data Buffers	The number of Hiperspace data buffers specified for the LSR pool. Source field: A08TDHBW
Successful look asides	The number of successful lookasides to data buffers for this LSR pool. Source field: A08TDBFF
Buffer reads	The number of read I/Os to the data buffers for this LSR pool. Source field: A08TDFRD
User initiated writes	The number of user-initiated buffer writes from the data buffers for this LSR pool. Source field: A08TDUIW
Non-user initiated writes	The number of non-user-initiated buffer writes from the data buffers for this LSR pool. Source field: A08TDNUW
Successful Hiperspace CREADS	The number of successful CREAD requests issued to transfer data from Hiperspace data buffers to virtual data buffers. Source field: A08TDCRS
Successful Hiperspace CWRITES	The number of successful CWRITE requests issued to transfer data from virtual data buffers to Hiperspace data buffers. Source field: A08TDCWS

Table 179. Fields in the LSR Pools Report (continued)

Field Heading	Description
Failing Hiperspace CREADS	The number of CREAD requests that failed. MVS had withdrawn the space and VSAM had to read data from DASD. Source field: A08TDCRF
Failing Hiperspace CWRITES	The number of CWRITE requests that failed. There was insufficient Hiperspace and VSAM had to write the data to DASD. Source field: A08TDCWF
Index Buffers	The number of index buffers specified for the LSR pool. Source field: A08TIBFN
Hiperspace Index Buffers	The number of Hiperspace index buffers specified for the LSR pool. Source field: A08TIHBW
Successful look asides	The number of successful lookasides to index buffers for this LSR pool. Source field: A08TIBFF
Buffer reads	The number of read I/Os to the index buffers for this LSR pool. Source field: A08TIFRD
User initiated writes	The number of user-initiated buffer writes from the index buffers for this LSR pool. Source field: A08TIUIW
Non-user initiated writes	The number of non-user-initiated buffer writes from the index buffers for this LSR pool. Source field: A08TINUW
Successful Hiperspace CREADS	The number of successful CREAD requests issued to transfer data from Hiperspace index buffers to virtual index buffers. Source field: A08TICRS
Successful Hiperspace CWRITES	The number of successful CWRITE requests issued to transfer data from virtual index buffers to Hiperspace index buffers. Source field: A08TICWS
Failing Hiperspace CREADS	The number of CREAD requests that failed. MVS had withdrawn the space and VSAM had to read index data from DASD. Source field: A08TICRF
Failing Hiperspace CWRITES	The number of CWRITE requests that failed. There was insufficient Hiperspace and VSAM had to write the index data to DASD. Source field: A08TICWF
Data Buffer Statistics	
Size	The size of the data buffers that are available to CICS. Source field: A08BKBSZ
Buffers	The number of buffers of each size available to CICS. Source field: A08BKBFN

Table 179. Fields in the LSR Pools Report (continued)

Field Heading	Description
Hiperspace Buffers	The number of Hiperspace buffers specified for the pool. Source field: A08BKHBN
Look Asides	The number of read requests that VSAM was able to satisfy without initiating an I/O operation; that is, the requested record, whether index or data, was already present in one of the buffer resident CIs. This means that no physical I/O had to be done to put the control interval in the buffer. The tuning methodology usually employed involves either increasing the number of buffers of a particular CI size until the ratio of lookasides to READs stops increasing significantly or, conversely, reducing the number of buffers until the ratio of lookasides to READs begins to drop significantly. For most data sets, successful lookaside hits on indexes are more likely. These statistics are obtained from VSAM and represent the activity after the pool was created. Source field: A08BKBFF
Reads	The number of I/O operations to the buffers that VSAM was required to initiate to satisfy the CICS application's activity. This figure represents failures to find the control interval in the buffers. These statistics are obtained from VSAM and represent the activity after the pool was created. Source field: A08BKFRD
User Writes	The number of user-initiated I/O WRITE operations from the buffers that VSAM was required to initiate to satisfy the CICS application's activity. These statistics are obtained from VSAM and represent the activity after the pool was created. Source field: A08BKUIW
Non-User Writes	The number of non-user initiated I/O WRITE operations from the buffers that VSAM was forced to initiate due to no buffers being available for reading the contents of a CI. These statistics are obtained from VSAM and represent the activity after the pool was created. Source field: A08BKNUW
Successful CREADS	The number of successful CREAD requests issued to transfer data from Hiperspace buffers to virtual buffers. Source field: A08BKCRS
Successful CWRITES	The number of successful CWRITE requests issued to transfer data from virtual buffers to Hiperspace buffers. Source field: A08BKCWS
Failing CREADS	The number of CREAD requests that failed. MVS had withdrawn the space and VSAM had to read data from DASD. Source field: A08BKCRF

Table 179. Fields in the LSR Pools Report (continued)

Field Heading	Description
Failing CWRITES	The number of CWRITE requests that failed. There was insufficient Hiperspace and VSAM had to write the data to DASD. Source field: A08BKCWF
Index Buffer Statistics	
Size	The size of the index data buffers that are available to CICS. Source field: A08IKBSZ
Buffers	The number of buffers of each size available to CICS. Source field: A08IKBFN
Hiperspace Buffers	The number of Hiperspace buffers specified for the pool. Source field: A08IKHBN
Look Asides	The number of read requests that VSAM was able to satisfy without initiating an I/O operation; that is, the requested index record was already present in one of the buffer resident CIs. This means that no physical I/O had to be done to put the control interval in the buffer. The tuning methodology usually employed involves either increasing the number of buffers of a particular CI size until the ratio of lookasides to READs stops increasing significantly or, conversely, reducing the number of buffers until the ratio of lookasides to READs begins to drop significantly. For most data sets, successful lookaside hits on indexes are more likely. These statistics are obtained from VSAM and represent the activity after the pool was created. Source field: A08IKBFF
Reads	The number of I/O operations to the buffers that VSAM was required to initiate to satisfy the CICS application's activity. This figure represents failures to find the control interval in the buffers. These statistics are obtained from VSAM and represent the activity after the pool was created. Source field: A08IKFRD
User Writes	The number of user-initiated I/O WRITE operations from the buffers that VSAM was required to initiate to satisfy the CICS application's activity. These statistics are obtained from VSAM and represent the activity after the pool was created. Source field: A08IKUIW
Non-User Writes	The number of non-user initiated I/O WRITE operations from the buffers that VSAM was forced to initiate due to no buffers being available for reading the contents of a CI. These statistics are obtained from VSAM and represent the activity after the pool was created. Source field: A08IKNUW
Successful CREADS	The number of successful CREAD requests issued to transfer data from Hiperspace buffers to virtual buffers. Source field: A08IKCRS

Table 179. Fields in the LSR Pools Report (continued)

Field Heading	Description
Successful CWRITES	The number of successful CWRITE requests issued to transfer data from virtual buffers to Hiperspace buffers. Source field: A08IKCWS
Failing CREADS	The number of CREAD requests that failed. MVS had withdrawn the space and VSAM had to read data from DASD. Source field: A08IKCRF
Failing CWRITES	The number of CWRITE requests that failed. There was insufficient Hiperspace and VSAM had to write the data to DASD. Source field: A08IKCWF

Files Report

Figure 66 shows the format of the Files Report. This report is produced using a combination of the EXEC CICS INQUIRE FILE and EXEC CICS COLLECT STATISTICS FILE commands. The statistics data is mapped by the DFHA17DS DSECT. The field headings and contents are described in Table 180.

```
Applid IYK2Z1V3  Sysid CJB3  Jobname CI07CJB3    Date 08/20/1998  Time 15:33:57                CICS 5.3.0                PAGE    72

Files

            Access     File     Remote   Remote  LSR           Data          CFDT      Table     Update
  Filename  Method     Type     Filename System  Pool  RLS     Table Type    Poolname  Name      Model
  ____________________________________________________________________________________________________

  ADMF      VSAM       KSDS                       1   No
  CSQKCDF   VSAM       KSDS                       1   No
  CSQ4FIL   VSAM       KSDS                       1   No
  DFHCMACD  VSAM       KSDS                       1   No
  DFHCSD    VSAM       KSDS                      No   No
  DFHDBFK   VSAM       KSDS                      No   No
  DFHLRQ    VSAM       KSDS                      No   No
  DFHRPCD   VSAM       KSDS                       1   No
  FILEA     VSAM       KSDS                       1   No
  RFSDIR1   VSAM       KSDS                       2   No
  RFSDIR2   VSAM       KSDS                       2   No
  RFSPOOL1  VSAM       KSDS                       2   No
  RFSPOOL2  VSAM       KSDS                       2   No
```

Figure 66. The Files Report

Table 180. Fields in the Files Report

Field Heading	Description
Files	
Filename	The name of the file. Source field: EXEC CICS INQUIRE FILE()
Access Method	Indicates the access method for this file. Source field: EXEC CICS INQUIRE FILE() ACCESSMETHOD(cvda)
File Type	Indicates how the records are organized in the data set that corresponds to this file. Source field: EXEC CICS INQUIRE FILE() TYPE(cvda)

Table 180. Fields in the Files Report (continued)

Field Heading	Description
Remote Filename	The name by which the file is known in the remote system. Source field: EXEC CICS INQUIRE FILE() REMOTENAME()
Remote System	The name of the CICS region in which the file is defined. Source field: EXEC CICS INQUIRE FILE() REMOTESYSTEM()
LSR Pool	The identity of the LSR pool defined for this file. "0" means that it is not defined in an LSR pool. Source field: EXEC CICS INQUIRE FILE() LSRPOOLID()
RLS	Indicates whether the file is to be opened in RLS mode. Source field: A17RLS
Data Table Type	The type of data table, coupling facility, CICS-maintained, user-maintained, or remote. Source field: EXEC CICS INQUIRE FILE() TABLE(cvda) REMOTETABLE(cvda)
CFDT Poolname	The name of the coupling facility data table pool in which the coupling facility data table resides. Source field: EXEC CICS INQUIRE FILE() CFDTPOOL()
Table Name	The couplinga facility data table name. Source field: EXEC CICS INQUIRE FILE() TABLENAME()
Update Model	Indicates the update model type for the coupling facility data table. Source field: EXEC CICS INQUIRE FILE() UPDATEMODEL(cvda)

File Requests Report

Figure 67 shows the format of the File Requests report. This report is produced using a combination of the EXEC CICS INQUIRE FILE and EXEC CICS COLLECT STATISTICS FILE commands. The field headings and contents are described in Table 181 on page 588.

```
Applid IYK2Z1V3  Sysid CJB3  Jobname CI07CJB3    Date 12/02/1998  Time 08:52:33                CICS 5.3.0                PAGE     3

Files - Requests

                Read     Get Update    Browse      Browse       Add        Update      Delete      Remote     RLS Req.
Filename     Requests    Requests     Requests     Updates    Requests Requests     Requests     Deletes    Timeouts

  ADMF              0           0            0           0           0           0           0           0           0
  BANKACCT          0           0            0           0           0           0           0           0           0
  CSQKCDF           0           0            0           0           0           0           0           0           0
  CSQ4FIL           0           0            0           0           0           0           0           0           0
  DFHCMACD          0           0            0           0           0           0           0           0           0
  DFHCSD            0           0            0           0           0           0           0           0           0
  DFHDBFK           0           0            0           0           0           0           0           0           0
  DFHLRQ            0           0            0           0           0           0           0           0           0
  DFHRPCD           0           0            0           0           0           0           0           0           0
  FILEA             0           0            0           0           0           0           0           0           0
  RFSDIR1           0           0            0           0           0           0           0           0           0
  RFSDIR2           0           0            0           0           0           0           0           0           0
  RFSPOOL1          0           0            0           0           0           0           0           0           0
  RFSPOOL2          0           0            0           0           0           0           0           0           0
```

Figure 67. The File Requests Report

Table 181. Fields in the File Requests Report

Field Heading	Description
Files - Requests	
Filename	The name of the file. Source field: EXEC CICS INQUIRE FILE()
Read Requests	The number of GET requests attempted against this file. Source field: A17DSRD
Get Update Requests	The number of GET UPDATE requests attempted against this file. Source field: A17DSGU
Browse Requests	The number of GETNEXT and GETPREV requests attempted against this file. Source field: A17DSBR
Browse Updates	The number of GETNEXT UPDATE and GETPREV UPDATE requests attempted against this file. Source field: A17DSBRU
Add Requests	The number of PUT requests attempted against this file. Source field: A17DSWRA
Update Requests	The number of PUT UPDATE requests attempted against this file. Source field: A17DSWRU
Delete Requests	The number of DELETE requests attempted against this local file. Source field: A17DSDEL
Remote Deletes	The number of DELETE requests for a VSAM file in a remote system. In systems connected by a CICS intercommunication (MRO or ISC) link, the statistics recorded for the remote files are a subset of those recorded for the files on the local system. Source field: A17RMDEL
RLS Req. Timeouts	The number of RLS file requests that timed out. Source field: A17RLSWT

Data Tables Reports

Figure 68 shows the format of the Data Tables Requests Report. This report is produced using a combination of the EXEC CICS INQUIRE FILE and EXEC CICS COLLECT STATISTICS FILE commands. The statistics data is mapped by the DFHA17DS DSECT. The field headings and contents are described in Table 182.

```
Applid IYK2Z1V3  Sysid CJB3  Jobname CI07CJB3    Date 01/28/1999  Time 11:07:52              CICS 5.3.0           PAGE     8
_____________________________________________________________________________________________________________________________

Data Tables - Requests
______________________

            Successful    Records    Adds via    Adds via      Adds       Adds       Rewrite      Delete       Read     Chng Resp/
  Filename     Reads     Not Found     Read         API      Rejected     Full      Requests     Requests     Retries   Lock Waits
  ___________________________________________________________________________________________________________________________

  F140BASE       0           0           1           0           0          0           0            0           0          0
  F150BASE       0           0           0           0           0          0           0            0           0          0
  F170BASE       0           0           0           0           0          0           0            0           0          0
  F270BASE       0           0           0           0           0          0           0            0           0          0
```

Figure 68. The Data Tables Report

Table 182. Fields in the Data Tables Requests Report

Field Heading	Description
Data Tables - Requests	
Filename	The name of the file. Source field: EXEC CICS INQUIRE FILE()
Successful Reads	The number of attempts to retrieve records from the table. Source field: A17DTRDS
Records Not Found	The number of times API READ requests were directed to the source data set because the record was not found in the table. Source field: A17DTRNF
Adds via Read	The number of records placed in the table by the loading process or as a result of API READ requests issued while loading was in progress. Source field: A17DTAVR
Adds via API	The number of attempts to add records to the table as a result of WRITE requests. Source field: A17DTADS
Adds Rejected	The number of records CICS attempted to add to the table which were rejected by the global user exit. Source field: A17DTARJ
Adds Full	The number of records CICS attempted to add to the table but was unable to do so because the table already contained the maximum number of records specified. Source field: A17DTATF
Rewrite Requests	The number of attempts to update records in the table as a result of REWRITE requests. Source field: A17DTRWS

Table 182. Fields in the Data Tables Requests Report (continued)

Field Heading	Description
Delete Requests	The number of attempts to delete records from the table as a result of DELETE requests. Source field: A17DTDLS
Read Retries	The total number of read retries, that is the number of times reads in an AOR had to be retried because the FOR changed the table during the read. Source field: A17DTRRS
Chng Resp/Lock Waits	For a CFDT that is using the locking model, records are locked when they are read for update. This count is the number of times it was necessary to WAIT for an already locked record. For a CFDT that is using the contention model, records are not locked when they are read for update. If a subsequent rewrite or delete request finds that the record has already changed a CHANGED response is returned. This count is the number of times that a CHANGED response was issued. Source field: A17DTCON

Figure 69 shows the format of the Data Tables Storage Report. The field headings and contents are described in Table 183.

```
Data Tables - Storage
Applid IYK2ZIV3  Sysid CJB3  Jobname CI07CJB3    Date 01/28/1999  Time 11:07:52              CICS 5.3.0              PAGE     9

 Data Tables - Storage:

      <------- Total ------> <------ Entries -----> <------- Index ------> <---- --- Data ------->
                  Current      Peak     Storage    Storage      Storage    Storage     Storage    Storage    Storage    Storage
  Filename Type   Records    Records   Allocated    In-Use     Allocated    In-Use    Allocated    In-Use   Allocated    In-Use

  F140BASE CICS       1          1         192          3           32          1           32          1         128          1
  F150BASE USER       1          1         192          3           32          1           32          1         128          1
  F170BASE USER       1          1         192          3           32          1           32          1         128          1
  F270BASE   CF       0          0           0          0            0          0            0          0           0          0
```

Figure 69. The Data Tables Storage Report

Table 183. Fields in the Data Tables Storage Report

Field Heading	Description
Data Tables - Storage	
Filename	The name of the file. Source field: EXEC CICS INQUIRE FILE()
Type	The type of data table, coupling facility, CICS-maintained or user-maintained. Source field: EXEC CICS INQUIRE FILE() TABLE(cvda)
Current Records	The current number of records in the data table. Source field: A17DTSIZ
Peak Records	The peak number of records in the data table. Source field: A17DTSHI
Total - Storage Allocated	The total amount of storage (kilobytes) in allocated for the data table. Source field: A17DTALT

Table 183. Fields in the Data Tables Storage Report (continued)

Field Heading	Description
Total - Storage In-Use	The total amount of storage (kilobytes) in use for the data table. Source field: A17DTUST
Entries - Storage Allocated	The total amount of storage (kilobytes) allocated for the record entry blocks. Source field: A17DTALE
Entries - Storage In-Use	The total amount of storage (kilobytes) in use for the record entry blocks. Source field: A17DTUSE
Index - Storage Allocated	The total amount of storage (kilobytes) allocated for the index. Source field: A17DTALI
Index - Storage In-Use	The total amount of storage (kilobytes) in use for the index. Source field: A17DTUSI
Data - Storage Allocated	The total amount of storage (kilobytes) allocated for the record data. Source field: A17DTALD
Data - Storage In-Use	The total amount of storage (kilobytes) in use for the record data. Source field: A17DTUSD

Coupling Facility Data Table Pools Report

Figure 70 shows the format of the Coupling facility data tables report. This report is produced using the EXEC CICS INQUIRE CFDTPOOL command. The field headings and contents are described in Table 184.

```
Applid IYK2Z1V3  Sysid CJB3  Jobname CI07CJB3    Date 02/26/1999  Time 08:18:14              CICS 5.3.0               PAGE     2
Coupling Facility Data Table Pools
Coupling Facility Data Table Pool .    CFPOOL1         Connection Status.. . . :  UNAVAILABLE
```

Figure 70. The Coupling Facility Data Table Pools Report

Table 184. Fields in the Coupling Facility Data Table Pools Report

Field Heading	Description
Coupling Facility Data Table Pools	
Coupling Facility Data Table Pool	The name of the coupling facility data table pool. Source field: EXEC CICS INQUIRE CFDTPOOL()
Connection Status	Indicates the connection status of the pool. Source field: EXEC CICS INQUIRE CFDTPOOL() CONNSTATUS(cvda)

Exit Programs Report

Figure 71 on page 592 shows the format of the Exit Programs Report. This report is produced using the EXEC CICS INQUIRE EXITPROGRAM command. The field headings and contents are described in Table 185 on page 592.

```
Applid IYK2Z1V3  Sysid CJB3  Jobname CI07CJB3    Date 08/20/1998  Time 15:33:57  CICS 5.3.0   PAGE    77


Exit Programs

                     <----- Global Area ----->   No.                                          Task    <-- Task Related User
  Program      Entry             Use    of    Program        Concurrency              Area    Task        Shut
    Name       Name    Length  Count  Exits  Status    API     Status  Length   start  EDF  down  Indoubt  SPI

  DFHEDP    DLI                    0       0      0  Started  Base  Quasi Rent              284    No    No    No   No Wa
  RMIIN     RMIIN                  0       0      1  Started  Base  Quasi Rent 0 No    No    No    No Wait   No
  TRUE      TRUE                   0       0      0  Stopped  Base  Quasi Rent 0 No    No    No    No Wait   No
```

Figure 71. The Exit Programs Report

Table 185. Fields in the Exit Programs Report

Field Heading	Description
Exit Programs	
Program Name	The program name of the program that has been enabled as an exit program using the EXEC CICS ENABLE command. Source field: EXEC CICS INQUIRE EXITPROGRAM()
Entry Name	The entry point name for this exit program. Source field: EXEC CICS INQUIRE EXITPROGRAM() ENTRYNAME()
Global Area Entry Name	The name of the exit program that owns the global work area associated with this exit program. Source field: EXEC CICS INQUIRE EXITPROGRAM() GAENTRYNAME()
Global Area Length	The length of the global work area associated with this exit program. Source field: EXEC CICS INQUIRE EXITPROGRAM() GALENGTH()
Global Area Use Count	The number of exit programs that are associated with the global work area owned by this exit program. Source field: EXEC CICS INQUIRE EXITPROGRAM() GAUSECOUNT()
Number of Exits	The number of global user exit points at which this exit program is enabled. Source field: EXEC CICS INQUIRE EXITPROGRAM() NUMEXITS()
Program Status	Indicates whether this exit program is available for execution. Source field: EXEC CICS INQUIRE EXITPROGRAM() STARTSTATUS(cvda)
API	Source field: EXEC CICS INQUIRE EXITPROGRAM() APIST(cvda)
Concurrency Status	Source field: EXEC CICS INQUIRE EXITPROGRAM() CONCURRENST(cvda)
Qualifier	The name of the qualifier specified for this exit program. Source field: EXEC CICS INQUIRE EXITPROGRAM() QUALIFIER()
Task Area Length	The length of the task local work area associated with this exit program. Source field: EXEC CICS INQUIRE EXITPROGRAM() TALENGTH()

Table 185. Fields in the Exit Programs Report (continued)

Field Heading	Description
TRUE - Taskstart	Indicates whether this exit program was enabled with the TASKSTART option. Source field: EXEC CICS INQUIRE EXITPROGRAM() TASKSTART(cvda)
TRUE - EDF	Indicates whether this exit program was enabled with the FORMATEDF option. Source field: EXEC CICS INQUIRE EXITPROGRAM() FORMATEDFST(cvda)
TRUE - Shutdown	Indicates whether this exit program was enabled with the SHUTDOWN option. Source field: EXEC CICS INQUIRE EXITPROGRAM() SHUTDOWNST(cvda)
TRUE Status - Indoubt	Indicates whether this exit program was enabled with the INDOUBTST option. Source field: EXEC CICS INQUIRE EXITPROGRAM() INDOUBTST(cvda)
TRUE Status - SPI	Indicates whether this exit program was enabled with the SPI option. Source field: EXEC CICS INQUIRE EXITPROGRAM() SPIST(cvda)

Global User Exits Report

Figure 72 shows the format of the Global User Exits Report. The field headings and contents are described in Table 186.

```
Applid IYK2Z1V3  Sysid CJB3  Jobname CI07CJB3    Date 08/20/1998  Time 15:33:57                CICS 5.3.0


Global User Exits

    Exit    Program    Entry    <------- Global Area ------->   Number   Program
    Name      Name      Name    Entry Name  Length  Use Count  of Exits  Status

  ____________________________________________________________________________

  XMNOUT    WLMMNOUT  WLMMNOUT                  0          0         1   Stopped

  XRMIIN    RMIIN     RMIIN                     0          0         1   Started
```

Figure 72. The Global User Exits Report

Table 186. Fields in the Global User Exits Report

Field Heading	Description
Global User Exits	
Exit Name	The name of the global user exit point. Source field: EXEC CICS INQUIRE EXITPROGRAM() EXIT()
Program Name	The name of the exit program enabled at this global user exit point. Source field: EXEC CICS INQUIRE EXITPROGRAM()
Entry Name	The name of the entry point for this exit program at this global user exit point. Source field: EXEC CICS INQUIRE EXITPROGRAM() ENTRYNAME()

Table 186. Fields in the Global User Exits Report (continued)

Field Heading	Description
Global Area Entry Name	The name of the exit program that owns the global work area associated with this exit program. Source field: EXEC CICS INQUIRE EXITPROGRAM() GAENTRYNAME()
Global Area Length	The length of the global work area for this exit program. Source field: EXEC CICS INQUIRE EXITPROGRAM() GALENGTH()
Global Area Use Count	The number of exit programs that are associated with the global work area owned by this exit program. Source field: EXEC CICS INQUIRE EXITPROGRAM() GAUSECOUNT()
Number of Exits	The number of global user exit points at which this exit program is enabled. Source field: EXEC CICS INQUIRE EXITPROGRAM() NUMEXITS()
Program Status	Indicates whether this exit program is available for execution. Source field: EXEC CICS INQUIRE EXITPROGRAM() STARTSTATUS(cvda)

DB2 Connection Report

Figure 73 on page 595 shows the format of the DB2 Connection Report. This report is produced using a combination of the EXEC CICS INQUIRE DB2CONN and EXEC CICS COLLECT STATISTICS DB2CONN commands. The statistics data is mapped by the DFHD2GDS DSECT.

Figure 73 on page 595 shows the format of the DB2 Connection Report. The field headings and contents are described in Table 187 on page 595

```
DB2 Connection

 DB2 Connection Name. . . . . . . . . . . :  RCTJT
 DB2 Sysid. . . . . . . . . . . . . . . . :  DB3A
 DB2 Release. . . . . . . . . . . . . . . :   3.1.0

 DB2 Connection Status. . . . . . . . . . :  CONNECTED          DB2 Connect Date and Time . . . :  02/06/1997  09:48:47.92429
 DB2 Connection Error . . . . . . . . . . :  SQLCODE
 DB2 Standby Mode . . . . . . . . . . . . :  CONNECT

 DB2 Pool Thread Plan Name. . . . . . . . :
 DB2 Pool Thread Dynamic Plan Exit Name . :  DSNCUEXT

 Pool Thread Authtype . . . . . . . . . . :  SIGNID
 Pool Thread Authid . . . . . . . . . . . :

 Command Thread Authtype. . . . . . . . . :  SIGNID
 Command Thread Authid. . . . . . . . . . :

 Signid for Pool/Entry/Command Threads. . :  AREDDIO

 Create Thread Error. . . . . . . . . . . :  N906               Message TD Queue 1. . . . . . . . . . . :  MSLG
 Protected Thread Purge Cycle . . . . . . :  00.30              Message TD Queue 2. . . . . . . . . . . :
 Deadlock Resolution. . . . . . . . . . . :  ROLLBACK          Message TD Queue 3. . . . . . . . . . . :
 Non-Terminal Intermediate Syncpoint. . . :  NORELEASE
 Pool Thread Wait Setting . . . . . . . . :  WAIT               Statistics TD Queue . . . . . . . . . . :  CSSL

 Pool Thread Priority . . . . . . . . . . :  HIGH               DB2 Accounting records by . . . . . . . :  TXID

 Current TCB Limit. . . . . . . . . . . . :          4
 Current number of TCBs . . . . . . . . . :          4
 Peak number of TCBs. . . . . . . . . . . :          4

 Current number of free TCBs. . . . . . . :          4

 Current number of tasks on TCB Readyq. . :          0
 Peak number of tasks on TCB Readyq . . . :          6

 Pool Thread Limit. . . . . . . . . . . . :          7          Number of Calls using Pool Threads. . . . . :          0
 Current number of Pool Threads . . . . . :          0          Number of Pool Thread Signons . . . . . . . :          0
 Peak number of Pool Threads. . . . . . . :          4          Number of Pool Thread Commits . . . . . . . :          0
 Number of Pool Thread Waits. . . . . . . :         53          Number of Pool Thread Aborts. . . . . . . . :          0
                                                               Number of Pool Thread Single Phase. . . . . :          0
 Current number of Pool Tasks . . . . . . :          0          Number of Pool Thread Reuses. . . . . . . . :          0
 Peak number of Pool Tasks. . . . . . . . :          0          Number of Pool Thread Terminates. . . . . . :         42
 Current Total number of Pool Tasks . . . :          0

 Current number of Tasks on Pool Readyq . :          0
 Peak number of Tasks on Pool Readyq. . . :          0

 Current number of DSNC Command threads . :          0          Number of DSNC Command Calls. . . . . . . . :          0
 Peak number of DSNC Command threads. . . :          0          Number of DSNC Command Signons. . . . . . . :          0
 DSNC Command Thread Limit. . . . . . . . :          2          Number of DSNC Command Thread Terminates. . :          0
                                                               Number of DSNC Command Thread Overflows . . :          0
```

Figure 73. The DB2 Connection Report

Table 187. Fields in the DB2 Connection Report

Field Heading	Description
DB2 Connection	
DB2 Connection Name	The name of the installed DB2CONN Source field: D2G-DB2CONN-NAME
DB2 Sysid	The name of the DB2 subsystem to which the CICS-DB2 attachment is connected or will connect. Source field: D2G-DB2-ID
DB2 Release	The version and release level of the DB2 subsystem to which CICS is currently connected. Source field: D2G-DB2-RELEASE

Table 187. Fields in the DB2 Connection Report (continued)

Field Heading	Description
DB2 Connection Status	The current status of the CICS-DB2 Connection. Source field: EXEC CICS INQUIRE DB2CONN CONNECTST
DB2 Connect Date and Time	The date and time that the CICS connected to the DB2 subsystem. Source field: D2G-CONNECT-TIME-LOCAL
DB2 Connection Error	specifies how CICS reports back to an application that issues an SQL request that CICS is not connected to DB2. Source field: EXEC CICS INQUIRE DB2CONN CONNECTERROR
DB2 Standby Mode	specifies the action to be taken by the CICS-DB2 attachment if the DB2 subsystem is not active when an attempt to start the connection from CICS to DB2 is made. Source field: EXEC CICS INQUIRE DB2CONN STANDBYMODE
DB2 Pool Thread Plan Name	The name of the plan used for the pool. Source field: D2G-POOL-PLAN-NAME
DB2 Pool Thread Dynamic Plan Exit Name	The name of the dynamic plan exit used for pool threads. Source field: D2G-POOL-PLANEXIT-NAME
Pool Thread Authtype	The type of id to be used for security checking when using pool threads. Source field: D2G-POOL-AUTHTYPE
Pool Thread Authid	The id to be used for security checking when using pool threads. Source field: D2G-POOL-AUTHID
Command Thread Authtype	The type of id to be used for security checking when using command threads. Source field: D2G-COMD-AUTHTYPE
Command Thread Authid	The id to be used for security checking when using command threads. Source field: D2G-COMD-AUTHID
Signid for Pool/Entry/Command Threads	The authorization id to be used by the CICS-DB2 attachment when signing on to DB2 for pool threads and DB2 entry threads when 'Pool Thread Authtype' is SIGNID and for command threads when 'Command Thread Authtype' is SIGNID. Source field: EXEC CICS INQUIRE DB2CONN SIGNID
Create Thread Error	specifies the action to be taken when a create thread error occurs. Source field: EXEC CICS INQUIRE DB2CONN THREADERROR

Table 187. Fields in the DB2 Connection Report (continued)

Field Heading	Description
Message TD Queue 1	The name of the first transient data queue to which unsolicited messages from the CICS-DB2 attacment will be sent. Source field: EXEC CICS INQUIRE DB2CONN MSGQUEUE1
Protected Thread Purge Cycle	The length of time (mm:ss) of the protected thread purge cycle. Source field: EXEC CICS INQUIRE DB2CONN PURGECYCLEM and PURGECYCLES
Message TD Queue 2	The name of the second transient data queue to which unsolicited messages from the CICS-DB2 attacment will be sent. Source field: EXEC CICS INQUIRE DB2CONN MSGQUEUE2
Deadlock Resolution	The action to be taken for a transaction using a pool thread that has been selected by DB2 as victim of a deadlock resolution. Source field: EXEC CICS INQUIRE DB2CONN DROLLBACK
Message TD Queue 3	The name of the third transient data queue to which unsolicited messages from the CICS-DB2 attacment will be sent. Source field: EXEC CICS INQUIRE DB2CONN MSGQUEUE3
Non-Terminal Intermediate Syncpoint	specifies whether non-terminal transactions will release threads for reuse at intermediate syncpoints. Source field: EXEC CICS INQUIRE DB2CONN NONTERMREL
Pool Thread Wait Setting	specifies whether transactions should wait for a pool thread or be abended if the number of active pool threads reaches the pool thread limit. Source field: D2G-POOL-THREADWAIT
Statistics TD Queue	The name of the transient data queue for the CICS-DB2 attachment statistics produced when the CICS-DB2 attachment is shut down. Source field: EXEC CICS INQUIRE DB2CONN STATSQUEUE
Pool Thread Priority	The priority of the pool thread subtasks relative to the CICS main task (QR TCB). Source field: D2G-POOL-PRIORITY
DB2 Accounting records by	specifies the frequency of DB2 accounting records to be produced for transactions using pool threads. Source field: D2G-POOL-ACCOUNTREC

Table 187. Fields in the DB2 Connection Report (continued)

Field Heading	Description
Current TCB Limit	The maximum number of subtask TCBs that can be attached to service DB2 requests. Source field: D2G-TCB-LIMIT
Current number of TCBs	The current number of subtask TCBs attached to service command, pool, and DB2Entry threads. Source field: D2G-TCB-CURRENT
Peak number of TCBs	The peak number of subtask TCBs attached to service command, pool, and DB2Entry threads. Source field: D2G-TCB-HWM
Current number of free TCBs	The current number of free TCBs; that is, subtask TCBs with no DB2 thread. Source field: D2G-TCB-FREE
Current number of tasks on TCB Readyq	The current number of CICS tasks waiting for a DB2 subtask TCB to become available. Source field: D2G-TCB-READYQ-CURRENT
Peak number of tasks on TCB Readyq	The peak number of CICS tasks that waited for a DB2 subtask TCB to become available. Source field: D2G-TCB-READYQ-PEAK
Pool Thread Limit	The maximum number of pool threads allowed. Source field: D2G-POOL-THREAD-LIMIT
Number of Calls using Pool Threads	The number of SQL calls made using pool threads. Source field: D2G-POOL-CALLS
Current number of Pool Threads	The current number of active pool threads. Source field: D2G-POOL-THREAD-CURRENT
Number of Pool Thread Signons	The number of DB2 sign-ons performed for pool threads. Source field: D2G-POOL-SIGNONS
Peak number of Pool Threads	The peak number of active pool threads. Source field: D2G-POOL-THREAD-HWM
Number of Pool Thread Commits	The number of two phase commits performed for units of work using pool threads. Source field: D2G-POOL-COMMITS
Number of Pool Thread Waits	The number of times all available threads in the pool were busy and a transaction had to wait for a thread to become available. This count includes transactions that overflow to the pool to acquire a thread and have to wait for a pool thread. Source field: D2G-POOL-THREAD-WAITS
Number of Pool Thread Aborts	The number of units of work using pool threads that were rolled back. Source field: D2G-POOL-ABORTS

Table 187. Fields in the DB2 Connection Report (continued)

Field Heading	Description
Number of Pool Thread Single Phase	The number of units of work using pool threads that used single-phase commit, either because they were read-only UOWs, or because DB2 was the only recoverable resource updated in the UOW. Source field: D2G-POOL-SINGLE-PHASE
Current number of Pool Tasks	The current number of CICS tasks using pool threads. Source field: D2G-POOL-TASK-CURRENT
Number of Pool Thread Reuses	The number of times CICS transactions using the pool were able to reuse an already created DB2 thread. This count includes transactions that overflow to the pool to acquire a thread and reuse an existing thread. P.Source field: D2G-POOL-THREAD-REUSE
Peak number of Pool Tasks	The peak number of CICS tasks using pool threads. Source field: D2G-POOL-TASK-HWM
Number of Pool Thread Terminates	The number of terminate thread requests made to DB2 for pool threads. This includes pool threads used by transactions that overflow to the pool. Source field: D2G-POOL-THREAD-TERM
Current Total number of Pool Tasks	The current total number of tasks that have used a pool thread. Source field: D2G-POOL-TASK-TOTAL + D2G-POOL-TASK-CURRENT
Current number of Tasks on Pool Readyq	The current number of CICS tasks waiting for a pool thread to become available. Source field: D2G-POOL-READYQ-CURRENT
Peak number of Tasks on Pool Readyq	The peak number of CICS tasks that waited for a pool thread to become available. Source field: D2G-POOL-READYQ-HWM
Current number of DSNC Command threads	The current number of active command threads servicing DB2 commands issued using the DSNC transaction. Source field: D2G-COMD-THREAD-CURRENT
Number of DSNC Command Calls	The number of DB2 commands issued using the DSNC transaction. Source field: D2G-COMD-CALLS
Peak number of DSNC Command threads	The peak number of command threads servicing DSNC DB2 commands. Source field: D2G-COMD-THREAD-HWM
Number of DSNC Command Signons	The number of DB2 sign-ons performed for DSNC DB2 commands. Source field: D2G-COMD-SIGNONS

Table 187. Fields in the DB2 Connection Report (continued)

Field Heading	Description
DSNC Command Thread Limit	The maximum number of command threads allowed for DSNC DB2 commands. Source field: D2G-COMD-THREAD-LIMIT
Number of DSNC Command Thread Terminates	The number of terminate thread requests made to DB2 for command threads. Source field: D2G-COMD-THREAD-TERM
Number of DSNC Command Thread Overflows	The number of times a DSNC DB2 command resulted in a pool thread being used because of the active number of command threads exceeding the command thread limit. Source field: D2G-COMD-THREAD-OVERF

DB2 Entries Report

Figure 74 shows the format of the DB2 Entries Report. This report is produced using a combination of the EXEC CICS INQUIRE DB2ENTRY and EXEC CICS COLLECT STATISTICS DB2ENTRY commands. The statistics data is mapped by the DFHD2RDS DSECT. The field headings and contents are described inTable 188.

```
Applid IYK2Z1V3  Sysid CJB3  Jobname CI07CJB3    Date 08/20/1998  Time 15:33:57              CICS 5.3.0               PAGE    80


DB2 Entries

  DB2Entry Name. . . . . . . . . . . . . . :  XC06               DB2Entry Status . . . . . . . . . . . . :  ENABLED
  DB2Entry Static Plan Name. . . . . . . . :  TESTC06            DB2Entry Disabled Action. . . . . . . . :  POOL
  DB2Entry Dynamic Plan Exit Name. . . . . :                     DB2Entry Deadlock Resolution. . . . . . :  ROLLBACK

  DB2Entry Authtype. . . . . . . . . . . . :  USERID             DB2Entry Accounting records by. . . . . :  UOW
  DB2Entry Authid. . . . . . . . . . . . . :
                                                                 Number of Calls using DB2Entry. . . . . . . :         456
  DB2Entry Thread Wait Setting . . . . . . :  POOL               Number of DB2Entry Signons. . . . . . . . . :          57
                                                                 Number of DB2Entry Commits. . . . . . . . . :           0
  DB2Entry Thread Priority . . . . . . . . :  HIGH               Number of DB2Entry Aborts . . . . . . . . . :           0
  DB2Entry Thread Limit. . . . . . . . . . :          1          Number of DB2Entry Single Phase . . . . . . :         114
  Current number of DB2Entry Threads . . . :          0          Number of DB2Entry Thread Reuses. . . . . . :           0
  Peak number of DB2Entry Threads. . . . . :          1          Number of DB2Entry Thread Terminates. . . . :          15
                                                                 Number of DB2Entry Thread Waits/Overflows . :          95
  DB2Entry Protected Thread Limit. . . . . . . . . :          0
  Current number of DB2Entry Protected Threads . . :          0
  Peak number of DB2Entry Protected Threads. . . . :          0

  Current number of DB2Entry Tasks . . . . . . . . :          0
  Peak number of DB2Entry Tasks. . . . . . . . . . :         10
  Current Total number of DB2Entry Tasks . . . . . :         57

  Current number of Tasks on DB2Entry Readyq . . . :          0
  Peak number of Tasks on DB2Entry Readyq. . . . . :          0

--------------------------------------------------------------------------------------------------------------------------------
```

Figure 74. The DB2 Entries Report

Table 188. Fields in the DB2 Entries Report

Field Heading	Description
DB2 Entries	
DB2Entry Name	The name of the installed DB2ENTRY. Source field: EXEC CICS INQUIRE DB2ENTRY

Table 188. Fields in the DB2 Entries Report (continued)

Field Heading	Description
DB2Entry Status	The current enabled status of this DB2Entry. Source field: EXEC CICS INQUIRE DB2ENTRY ENABLESTATUS
DB2Entry Static Plan Name	The name of the plan to be used for this DB2Entry. Source field: D2R-PLAN-NAME
DB2Entry Disabled Action	The action to be taken for new CICS tasks that attempt to use this DB2entry when it is disabled or being disabled. Source field: EXEC CICS INQUIRE DB2ENTRY DISABLEDACT
DB2Entry Dynamic Plan Exit Name	The name of the dynamic plan exit used by this DB2Entry. Source field: D2R-PLANEXIT-NAME
DB2Entry Deadlock Resolution	The action to be taken for a transaction using a thread from this DB2Entry that has been selected by DB2 as a victim of deadlock resolution. Source field: EXEC CICS INQUIRE DB2ENTRY DROLLBACK
DB2Entry Authtype	The type of id to be used for security checking for threads of this DB2Entry. Source field: D2R-AUTHTYPE
DB2Entry Accounting records by	specifies the frequency of DB2 accounting records to be produced for transactions using this DB2Entry. Source field: D2R-ACCOUNTREC
DB2Entry Authid	The id to be used for security checking for threads of this DB2Entry. Source field: D2R-AUTHID
Number of Calls using DB2Entry	The number of SQL calls made using a thread from this DB2Entry. Source field: D2R-CALLS
DB2Entry Thread Wait Setting	specifies whether or not transactions should wait for a DB2Entry thread, be abended, or overflow to the pool should the number of active threads reach the thread limit for this DB2Entry. Source field: D2R-THREADWAIT
Number of DB2Entry Signons	The number of DB2 sign-ons performed for threads of this DB2Entry. Source field: D2R-SIGNONS
Number of DB2Entry Commits	The number of two-phase commits performed for Units of work using threads from this DB2ENTRY. Source field: D2R-COMMITS

Table 188. Fields in the DB2 Entries Report (continued)

Field Heading	Description
DB2Entry Thread Priority	The priority of the thread subtasks for this DB2Entry relative to the CICS main task (QR TCB). Source field: D2R-PRIORITY
Number of DB2Entry Aborts	The number of units of work using threads from this DB2ENTRY that were rolled back. Source field: D2R-ABORTS
DB2Entry Thread Limit	The maximum number of threads allowed for this DB2Entry. Source field: D2R-THREAD-LIMIT
Number of DB2Entry Single Phase	The number of units of work using threads from this DB2Entry that used single-phase commit, either because they were read-only UOWs, or because DB2 was the only recoverable resource updated in the UOW. Source field: D2R-SINGLE-PHASE
Current number of DB2Entry Threads	The current number of active threads using this DB2Entry. Source field: D2R-THREAD-CURRENT
Number of DB2Entry Thread Reuses	The number of times CICS transactions using this DB2Entry were able to reuse an already created DB2 thread. Source field: D2R-THREAD-REUSE
Peak number of DB2Entry Threads	The peak number of active threads for this DB2Entry. Source field: D2R-THREAD-HWM
Number of DB2Entry Thread Terminates	The number of terminate thread requests made for threads for this DB2Entry. Source field: D2R-THREAD-TERM
Number of DB2Entry Thread Waits/Overflows	The number of times all available threads for this DB2Entry were busy and a transaction had to wait for a thread to become available or 'overflow' to the pool and use a pool thread. Source field: D2R-THREAD-WAIT-OR-OVERFL
DB2Entry Protected Thread Limit	The maximum number of protected threads allowed for this DB2Entry. Source field: D2R-PTHREAD-LIMIT
Current number of DB2Entry Protected Threads	The current number of inactive threads of this DB2Entry that are protected. Source field: D2R-PTHREAD-CURRENT
Peak number of DB2Entry Protected Threads	The peak number of inactive threads of this DB2Entry that were protected. Source field: D2R-PTHREAD-HWM

Table 188. Fields in the DB2 Entries Report (continued)

Field Heading	Description
Current number of DB2Entry Tasks	The current number of CICS tasks using this DB2Entry. Source field: D2R-TASK-CURRENT
Peak number of DB2Entry Tasks	The peak number of CICS tasks using this DB2Entry. Source field: D2R-TASK-HWM
Current Total number of DB2Entry Tasks	The current total number of tasks that have used this DB2Entry. Source field: D2R-TASK-TOTAL + D2R-TASK-CURRENT
Current number of Tasks on DB2Entry Readyq	The current number of CICS tasks waiting for a thread to become available for this DB2Entry. Source field: D2R-READYQ-CURRENT
Peak number of Tasks on DB2Entry Readyq	The peak number of CICS tasks that waited for a thread to become available for this DB2Entry. Source field: D2R-READYQ-HWM

Enqueue Manager Report

Figure 75 on page 604 shows the format of the Enqueue Manager Report. This report is produced using the EXEC CICS COLLECT STATISTICS ENQUEUE command. The statistics data is mapped by the DFHNQGDS DSECT. The field headings and contents are described in Table 189 on page 604.

```
Applid IYK2Z1V3  Sysid CJB3  Jobname CI07CJB3    Date 08/20/1998  Time 15:33:57 CICS 5.3.0   PAGE    81


Enqueue Manager  ENQueue poolname . . . . . . . . . . . . :       DISPATCH

  ENQs issued. . . . . . . . . . . . . . . . :              0
  ENQs waited. . . . . . . . . . . . . . . . :              0
  ENQueue waiting time . . . . . . . . . . . :  00:00:00.00000
  Average Enqueue wait time. . . . . . . . . :  00:00:00.00000

  Current ENQs waiting . . . . . . . . . . . :              0
  Current ENQueue waiting time . . . . . . . :  00:00:00.00000

  Sysplex ENQs waited. . . . . . . . . . . . :              0
  Sysplex ENQueue waiting time . . . . . . . :  00:00:00.00000
  Average Sysplex Enqueue wait time. . . . . :  00:00:00.00000

  Current Sysplex ENQs waiting . . . . . . . :              0
  Current Sysplex ENQueue waiting time . . . :  00:00:00.00000
  Total ENQs retained. . . . . . . . . . . . :              0
  Enqueue retention time . . . . . . . . . . :  00:00:00.00000
  Average Enqueue retention time . . . . . . :  00:00:00.00000

  Current ENQs retained. . . . . . . . . . . :              0
  Current Enqueue retention time . . . . . . :  00:00:00.00000
  Current Average Enqueue retention time . . :  00:00:00.00000

  Enqueues Rejected - Enqbusy. . . . . . . . :              0
  Enqueues Rejected - Retained . . . . . . . :              0

  Waiting Enqueues - Rejected Retained . . . :              0
  Waiting Enqueues Purged - Operator . . . . :              0
  Waiting Enqueues Purged - Timeout. . . . . :              0
```

Figure 75. The Enqueue Manager Report

Table 189. Fields in the Enqueue Manager Report

Field Heading	Description
Enqueue Manager	
ENQueue poolname	The enqueue pool name. Source field: NQGPOOL
ENQs issued	The number of enqueues issued. Source field: NQGTNQSI
ENQs waited	The number of enqueues that waited. Source field: NQGTNQSW
ENQueue waiting time	The enqueue waiting time. Source field: NQGTNQWT
Average Enqueue wait time	The average enqueue wait time. Source field: NQGTNQWT / NQGTNQSW
Current ENQs waiting	The current number of ENQs waiting. Source field: NQGCNQSW
Current ENQueue waiting time	 Source field: NQGCNQWT

Table 189. Fields in the Enqueue Manager Report (continued)

Field Heading	Description
Sysplex ENQs waited	Source field: NQGGNQSW
Sysplex ENQueue waiting time	Source field: NQGGNQWT
Average Sysplex Enqueue wait time	Source field: NQGGNQWT / NQGGNQSW
Current Sysplex ENQs waiting	Source field: NQGSNQSW
Current Sysplex ENQueue waiting time	Source field: NQGSNQWT
Total ENQs retained	The total number of enqueues retained. Source field: NQGTNQSR
Enqueue retention time	The enqueue retention time. Source field: NQGTNQRT
Average Enqueue retention time	The average enqueue retention time. Source field: NQGTNQRT / NQGTNQSR
Current ENQs retained	The current number of enqueues retained. Source field: NQGCNQSR
Current Enqueue retention time	The current enqueue retention time. Source field: NQGCNQRT
Current Average Enqueue retention time	The current average enqueue retention time. Source field: NQGCNQRT / NQGCNQSR
Enqueues Rejected - Enqbusy	The number of enqueues rejected immediately - ENQBUSY. Source field: NQGTIRJB
Enqueues Rejected - Retained	The number of immediately rejected retained enqueues. Source field: NQGTIRJR
Waiting Enqueues - Rejected Retained	The number of retained enqueues awaiting rejection. Source field: NQGTWRJR
Waiting Enqueues Purged - Operator	The number of enqueues awaiting rejection because of operator intervention. Source field: NQGTWPOP
Waiting Enqueues Purged - Timeout	The number of enqueues awaiting rejection because of timeout. Source field: NQGTWPTO

Recovery Manager Report

Figure 76 shows the format of the Recovery Manager Report. This report is produced using the EXEC CICS COLLECT STATISTICS RECOVERY command. The statistics data is mapped by the DFHRMGDS DSECT. The field headings and contents are described in Table 190.

```
Applid IYK2Z1V3  Sysid CJB3  Jobname CI07CJB3    Date 08/20/1998  Time 15:33:57   CICS 5.3.0     PAGE    99


Recovery Manager

  Number of Syncpoints forward . . . . . :          16
  Number of Syncpoints backward. . . . . :           0
  Number of Resynchronisations . . . . . :           0


  Total UOWs shunted for indoubt failure . . . . . :               0
  Total time UOWs shunted for indoubt failure. . . . . :  00:00:00.00000

  Current UOWs shunted for indoubt failure . . . . . :             0
  Total time current UOWs shunted for indoubt failure. . . . . :  00:00:00.00000

  Total UOWs shunted for commit/backout failure. . . . . :         0
  Total time UOWs shunted for commit/backout failure . . . . . :  00:00:00.00000

  Current UOWs shunted for commit/backout failure. . . . . :       0
  Total time current UOWs shunted for commit/backout failure . . :  00:00:00.00000


  Indoubt Action Forced by Trandef . . . . . :           0
  Indoubt Action Forced by Timeout . . . . . :           0
  Indoubt Action Forced by No Wait . . . . . :           0
  Indoubt Action Forced by Operator. . . . . :           0
  Indoubt Action Forced by Other . . . . . :             0

  Indoubt Action Forced by TD Queues . . . . . :         0
  Indoubt Action Forced by LU61 Connections. . :         0
  Indoubt Action Forced by MRO Connections . . :         0
  Indoubt Action Forced by RMI Exits . . . . . :         0
  Indoubt Action Forced by Other . . . . . :             0

  Number of Indoubt Action Mismatches. . . . . :         0
```

Figure 76. The Recovery Manager Report

Table 190. Fields in the Recovery Manager Report

Field Heading	Description
Recovery Manager	
Number of Syncpoints forward	The number of syncpoints issued. Source field: RMGSYFWD
Number of Syncpoints backward	The number of syncpoint rollbacks issued. Source field: RMGSYBWD
Number of Resynchronizations	The number of resyncs issued. Source field: RMGRESYN

Table 190. Fields in the Recovery Manager Report (continued)

Field Heading	Description
Total UOWs shunted for indoubt failure	The total number of UOWs shunted for indoubt failure. Source field: RMGTSHIN
Total time UOWs shunted for indoubt failure	The total time UOWs were shunted for indoubt failure. Source field: RMGTSHTI
Current UOWs shunted for indoubt failure	The current number of UOWs shunted for indoubt failure. Source field: RMGCSHIN
Total time current UOWs shunted for indoubt failure	The total time for the current UOWs shunted for indoubt failure. Source field: RMGCSHTI
Total UOWs shunted for commit/backout failure	The total number of UOWs shunted for commit/backout failure. Source field: RMGTSHRO
Total time UOWs shunted for commit/backout failure	The total time UOWs were shunted for commit/backout failure. Source field: RMGTSHTR
Current UOWs shunted for commit/backout failure	The current number of UOWs shunted for commit/backout failure. Source field: RMGCSHRO
Total time current UOWs shunted for commit/backout failure	The total time for the current UOWs shunted for commit/backout failure. Source field: RMGCSHTR
Indoubt Action Forced by Trandef	The number of forced indoubt action resolutions due to the transaction definition specifying that it cannot support indoubt waiting. Source field: RMGIAFTR
Indoubt Action Forced by Timeout	The number of forced indoubt action resolutions due to the indoubt wait timing out. Source field: RMGIAFTI
Indoubt Action Forced by No Wait	The number of forced indoubt action resolutions due to a recoverable resource or resource manager coordinator being unable to support indoubt waiting. Source field: RMGIAFNW
Indoubt Action Forced by Operator	The number of forced indoubt action resolutions due to the operator (CEMT or SPI command) cancelling the wait for indoubt resolution. Source field: RMGIAFOP
Indoubt Action Forced by Other	The number of forced indoubt action resolutions due to the all other reason other than those split out above. Source field: RMGIAFOT

Table 190. Fields in the Recovery Manager Report (continued)

Field Heading	Description
Indoubt Action Forced by TD Queues	The number of forced indoubt action resolutions due to a recoverable resource or resource manager coordinator being unable to support indoubt waiting. Source field: RMGNWTD
Indoubt Action Forced by LU61 Connections	The number of forced indoubt action resolutions due to a recoverable resource or resource manager coordinator being unable to support indoubt waiting. Source field: RMGNW61
Indoubt Action Forced by MRO Connections	The number of forced indoubt action resolutions due to a recoverable resource or resource manager coordinator being unable to support indoubt waiting. Source field: RMGNWMRO
Indoubt Action Forced by RMI Exits	The number of forced indoubt action resolutions due to a recoverable resource or resource manager coordinator being unable to support indoubt waiting. Source field: RMGNWRMI
Indoubt Action Forced by Other	The number of forced indoubt action resolutions due to a recoverable resource or resource manager coordinator being unable to support indoubt waiting. Source field: RMGNWOTH
Number of Indoubt Action Mismatches	The number of forced indoubt action resolutions that a participating resource manager coordinator resolved in the opposite way to CICS. Source field: RMGIAMIS

Page Index Report

Figure 77 on page 609 shows the format of the Page Index Report.

Page Index

'N/S' indicates that the statistics report was not selected for printing

Figure 77. The Page Index Report

Appendix F. MVS and CICS virtual storage

Important note
This appendix provides some data relating to other products installed with CICS. This information was valid when this book was written, but no guarantee can be given that the information will stay unchanged, either for other products or for CICS itself. You should always check the information with your local IBM Systems Center.

This appendix describes:

- Major elements of "MVS storage"
- Major elements of "The CICS private area" on page 614
- Contents of "The dynamic storage areas" on page 620.

Most of the CICS storage areas are moved above the line, and it is necessary to have some detailed knowledge of the components that make up the total address space in order to determine what is really required.

Furthermore, it is important to understand the implications of the value of MXT (maximum tasks) on the storage use within the CICS address space. For a given region size, a high MXT value reduces the storage available for other users in the dynamic storage areas.

MVS storage

There are four major elements of virtual storage within MVS. Each storage area is duplicated above 16MB.

- The common area below 16MB
- The private area below 16MB
- The extended common area above 16MB
- The extended private area above 16MB.

The MVS common area

The common areas contain the following elements:

- MVS/ESA nucleus and extended nucleus
- System queue area (SQA and ESQA)
- Link pack areas (PLPA, MLPA, and CLPA)
- Common system area (CSA and ECSA)
- Prefixed storage area (PSA).

All the elements of the common area above are duplicated above 16MB line with the exception of the PSA.

MVS nucleus and MVS extended nucleus

This is a static area containing the nucleus load module and extension to the nucleus. Although its size is variable depending on an installation's configuration, it cannot change without a re-IPL of MVS. The nucleus area below the 16MB line does not include page frame table entries, and the size of the nucleus area is

rounded up to a 4KB boundary. In addition, the nucleus area is positioned at the top of the 16MB map while the extended nucleus is positioned just above 16MB.

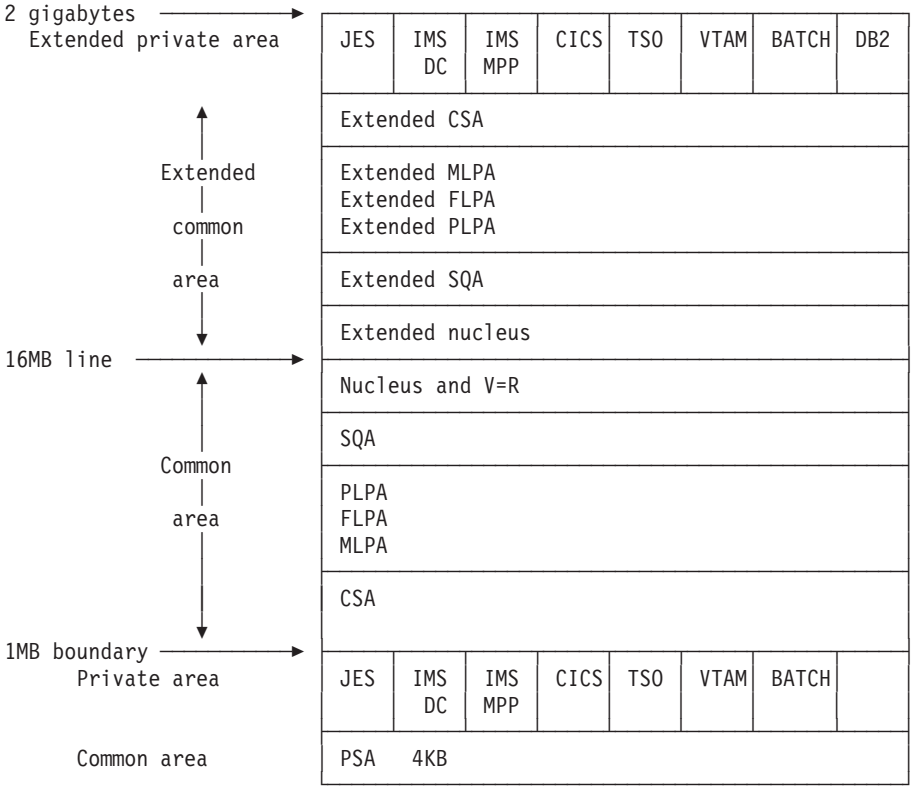


Figure 78. Virtual storage map for MVS/ESA

System queue area (SQA) and extended system queue area (ESQA)

This area contains tables and queues relating to the entire system. Its contents are highly dependent on configuration and job requirements at an installation. The total amount of virtual storage, number of private virtual storage address spaces, and size of the installation performance specification table are some of the factors that affect the system's use of SQA. The size of the initial allocation of SQA is rounded up to a 64KB boundary, though SQA may expand into the common system area (CSA) in increments of 4KB.

If the SQA is overallocated, the virtual storage is permanently wasted. If it is underallocated, it expands into CSA, if required. In a storage constrained system, it is better to be slightly underallocated. This can be determined by looking at the amount of free storage. If the extended SQA is underallocated, it expands into the extended CSA. When both the extended SQA and extended CSA are used up, the system allocates storage from SQA and CSA below the 16MB line. The allocation of this storage could eventually lead to a system failure, so it is better to overallocate extended SQA and extended CSA.

Link pack area (LPA) and extended link pack area (ELPA)

The link pack area (LPA) contains all the common reentrant modules shared by the system, and exists to provide:

- Economy of real storage by sharing one copy of the modules
- Protection: LPA code cannot be overwritten even by key 0 programs

- Reduced pathlength, because modules can be branched to.

It has been established that a 2MB LPA is sufficient for MVS when using CICS with MRO or ISC, that is, the size of an unmodified LPA as shipped by IBM. If it is larger, there are load modules in the LPA that may be of no benefit to CICS. There may be SORT, COBOL, ISPF, and other modules that are benefiting batch and TSO users. You have to evaluate if the benefits you are getting are worth the virtual storage that they use. If modules are removed, be sure to determine if the regions they run in need to be increased in size to accommodate them.

The pageable link pack area (PLPA) contains supervisor call routines (SVCs), access methods, and other read-only system programs along with read-only re-enterable user programs selected by an installation to be shared among users of the system. Optional functions or devices selected by an installation during system generation add additional modules to the PLPA.

The modified link pack area (MLPA) contains modules that are an extension to the PLPA. The MLPA may be changed at IPL without requiring the create link pack area (CLPA) option at IPL to change modules in the PLPA.

MVS/ESA common system area (CSA) and extended common system area (ECSA)

These areas contain pageable system data areas that are addressable by all active virtual storage address spaces. They contain, for example, buffers or executable modules for IMS/ESA, ACF/VTAM, JES3, and so on. Also present are control blocks used to define subsystems and provide working storage for such areas as TSO input/output control (TIOC), event notification facility (ENF), message processing facility (MPF), and so on. As system configuration and activity increase, the storage requirements also increase.

CICS uses the ECSA only if IMS/ESA or MRO is used. Even in this case, this use is only for control blocks and not for data transfer. If cross-memory facilities are being used, the ECSA usage is limited to 20 bytes per session and 1KB per address space participating in MRO. The amount of storage used by CICS MRO is given in the DFHIR3794 message issued to the CSMT destination at termination.

For static systems, the amount of unallocated CSA should be around 10% of the total allocated CSA; for dynamic systems, a value of 20% is desirable. Unlike the SQA, if CSA is depleted there is no place for it to expand into and a re-IPL can very possibly be required.

Prefixed storage area (PSA)

The PSA contains processor-dependent status information such as program status words (PSWs). There is one PSA per processor; however, all of them map to virtual storage locations 0–4KB as seen by that particular processor. MVS/ESA treats this as a separate area; there is no PSA in the extended common area.

Private area and extended private area

The private area contains:

- A local system queue area (LSQA)
- A scheduler work area (SWA)
- Subpools 229 and 230 (the requestor protect key area)
- A 16KB system region area (used by the initiator)
- A private user region for running programs and storing data.

Except for the 16KB system region area, each storage area in the private area has a counterpart in the extended private area.

The portion of the user's private area within each virtual address space that is available to the user's application program, is referred to as its **region**. The private area user region may be any size up to the size of the entire private area (from the top end of the PSA to the beginning, or bottom end, of the CSA) ***minus*** the size of LSQA, SWA, subpools 229 and 230, and the system region: for example, 220KB. (It is recommended that the region be 420KB less to allow for RTM processing.)

The segment sizes are one megabyte, therefore CSA is rounded up to the nearest megabyte. The private area is in increments of one megabyte. For more information, see "The CICS private area".

The CICS private area

The CICS private area has both static and dynamic storage requirements. The static areas are set at initialization time and do not vary over the execution of that address space. The dynamic areas increase or decrease their allocations as the needs of the address space vary, such as when data sets are opened and closed, and VTAM inbound messages being queued.

This section describes the major components of the CICS address space. In CICS Transaction Server for OS/390 Release 3 there are eight dynamic storage areas. They are:

The user DSA (UDSA)
The user-key storage area for all user-key task-lifetime storage below the 16MB boundary.

The read-only DSA (RDSA)
The key-0 storage area for all reentrant programs and tables below the 16MB boundary.

The shared DSA (SDSA)
The user-key storage area for any non-reentrant user-key RMODE(24) programs, and also for any storage obtained by programs issuing CICS GETMAIN commands for storage below the 16MB boundary with the SHARED option.

The CICS DSA (CDSA)
The CICS-key storage area for all non-reentrant CICS-key RMODE(24) programs, all CICS-key task-lifetime storage below the 16MB boundary, and for CICS control blocks that reside below the 16MB boundary.

The extended user DSA (EUDSA)
The user-key storage area for all user-key task-lifetime storage above the 16MB boundary.

The extended read-only DSA (ERDSA)
The key-0 storage area for all reentrant programs and tables above the 16MB boundary.

The extended shared DSA (ESDSA)
The user-key storage area for any non-reentrant user-key RMODE(ANY) programs, and also for any storage obtained by programs issuing CICS GETMAIN commands for storage above the 16MB boundary with the SHARED option.

The extended CICS DSA (ECDSA).
The CICS-key storage area for all non-reentrant CICS-key RMODE(ANY) programs, all CICS-key task-lifetime storage above the 16MB boundary, and CICS control blocks that reside above the 16MB boundary.

Figure 79 shows an outline of the areas involved in the private area. The three main areas are HPA, MVS storage, and the CICS region. The exact location of the free and allocated storage may vary depending on the activity and the sequence of the GETMAIN/FREEMAIN requests.

Additional MVS storage may be required by CICS for kernel stack segments for CICS system tasks—this is the CICS kernel.

Note: The CICS extended private area is conceptually the same as the CICS private area except that there is no system region. All the other areas have equivalent areas above the 16MB line.

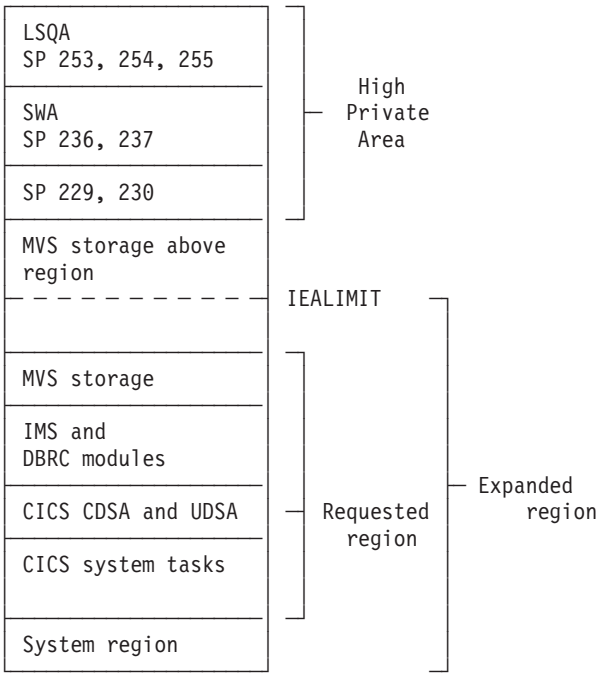


Figure 79. CICS private area immediately after system initialization

High private area

This area consists of four areas:

- LSQA
- SWA
- Subpool 229
- Subpool 230.

The area at the high end of the address space is not specifically used by CICS, but contains information and control blocks that are needed by the operating system to support the region and its requirements.

The usual size of the high private area varies with the number of job control statements, messages to the system log, and number of opened data sets.

The total space used in this area is reported in the IEF374I message in the field labeled "SYS=nnnnK" at jobstep termination. There is a second "SYS=nnnnK" which is issued which refers to the high private area above 16MB. This information is also reported in the sample statistics program, DFH0STAT.

Very little can be done to reduce the size of this area, with the possible exception of subpool 229. This is where VTAM stores inbound messages when CICS does not have an open receive issued to VTAM. The best way to determine if this is happening is to use CICS statistics (see "VTAM statistics" on page 494) obtained following CICS shutdown. Compare the maximum number of RPLs posted, which is found in the shutdown statistics, with the RAPOOL value in the SIT. If they are equal, there is a very good chance that subpool 229 is being used to stage messages, and the RAPOOL value should be increased.

The way in which the storage within the high private area is used can cause an S80A abend in some situations. There are at least two considerations:

1. **The use of MVS subpools 229 and 230 by access methods such as VTAM**: VTAM and VSAM may find insufficient storage for a request for subpools 229 and 230. Their requests are conditional and so should not cause an S80A abend of the job step (for example, CICS).
2. **The MVS operating system itself, relative to use of LSQA and SWA storage during job-step initiation**: The MVS initiator's use of LSQA and SWA storage may vary, depending on whether CICS was started using an MVS START command or started as a job step as part of already existing initiator and address space. Starting CICS with an MVS START command is better for minimizing fragmentation within the space above the region boundary. If CICS is a job step initiated in a previously started initiator's address space, the manner in which LSQA and SWA storage is allocated may reduce the apparently available virtual storage because of increased fragmentation.

Storage above the region boundary must be available for use by the MVS initiator (LSQA and SWA) and the access method (subpools 229 and 230).

Consider initiating CICS using an MVS START command, to minimize fragmentation of the space above your specified region size. This may avoid S80A abends by more effective use of the available storage.

Your choice of sizes for the MVS nucleus, MVS common system area, and CICS region influences the amount of storage available for LSQA, SWA, and subpools 229 and 230. It is unlikely that the sizes and boundaries for the MVS nucleus and common system area can easily be changed. To create more space for the LSQA, SWA, and subpools 229 and 230, you may need to ***decrease*** the region size.

Local system queue area (LSQA)

This area generally contains the control blocks for storage and contents supervision. Depending on the release level of the operating system, it may contain subpools 233, 234, 235, 253, 254, and 255.

The total size of LSQA is difficult to calculate because it depends on the number of loaded programs, tasks, and the number and size of the other subpools in the address space. As a guideline, the LSQA area usually runs between 40KB and 170KB depending on the complexity of the rest of the CICS address space.

The storage control blocks define the storage subpools within the private area, describing the free and allocated areas within those subpools. They may consist of such things as subpool queue elements (SPQEs), descriptor queue elements (DQEs), and free queue elements (FQEs).

The contents management control blocks define the tasks and programs within the address space such as task control blocks (TCBs), the various forms of request blocks (RBs), contents directory elements (CDEs), and many more.

CICS DBCTL requires LSQA storage for DBCTL threads. Allow 9KB for every DBCTL thread, up to the MAXTHRED value.

Scheduler work area (SWA)

This area is made up of subpools 236 and 237, which contain information about the job and step itself. Almost anything that appears in the job stream for the step creates some kind of control block here.

Generally, this area can be considered to increase with an increase in the number of DD statements. The distribution of storage in subpools 236 and 237 varies with the operating system release and whether dynamic allocation is used. The total amount of storage in these subpools is around 100KB to 150KB to start with, and it increases by about 1KB to 1.5KB per allocated data set.

A subset of SWA control blocks can, optionally, reside above 16MB. JES2 and JES3 have parameters that control this. If this needs to be done on an individual job basis, the SMF exit, IEFUJV, can be used.

Subpool 229

This subpool is used primarily for the staging of messages. JES uses this area for messages to be printed on the system log and JCL messages as well as SYSIN/SYSOUT buffers. Generally, a value of 40 to 100 KB is acceptable, depending on the number of SYSIN and SYSOUT data sets and the number of messages in the system log.

Subpool 230

This subpool is used by VTAM for inbound message assembly for segmented messages. Data management keeps data extent blocks (DEBs) here for any opened data set.

Generally, the size of subpool 230 increases as the number of opened data sets increases. Starting with an initial value of 40KB to 50KB, allow 300 to 400® per opened data set.

CICS DBCTL requires subpool 230 storage for DBCTL threads. Allow 3KB for every DBCTL thread, up to the MAXTHRED value.

MVS storage above region

This is the storage that is left between the top of the region and the bottom of the high private area. We recommend that 200KB to 300KB of free storage be maintained to allow for use by the termination routines in the case of an abend.

If this free storage is not enough for recovery termination management (RTM) processing, the address space may be terminated with a S40D abend that produces no dump.

This area can be very dynamic. As the high private area grows, it extends down into this area, and the CICS region may extend upward into this area up to the value specified in IEALIMIT.

The CICS region

The total storage for a CICS region is reported by the IEF374I message in the "VIRT=nnnK" portion for most address spaces. The equivalent message above the 16MB line is "EXT=nnnK". The sample statistics program, DFH0STAT, produces a report with this information. CICS formatted dumps and some specialized monitors may be useful to determine the sizes of the various components mentioned below. CICS statistics reports contain useful information about the subpools of the dynamic storage areas.

CICS virtual storage

CICS virtual storage requirements can be divided into:

- ***MVS Storage***: storage available to the operating system to perform region related services.
- ***Dynamic storage area***: the requirements of CICS, access methods, and applications.

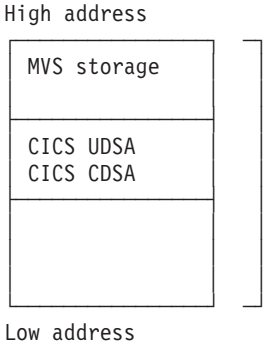


Figure 80. CICS region layout

MVS storage

The part of the CICS address space called MVS storage is the storage available to the MVS operating system to perform region-related services in response to an operating system macro or SVC issued by the region. For example, operating system components such as VSAM, DL/I, or DB2, issue MVS GETMAIN requests to obtain storage in which to build control blocks and these requests are met from MVS storage.

This is the amount of storage that remains after the dynamic storage areas and other CICS storage requirements have been met. The size of this area depends on MVS GETMAIN requirements during the execution of CICS. Opening files is the major contributor to usage of this area.

MVS storage is used to contain control blocks and data areas needed to open data sets or do other operating system functions, and program modules for the access method routines not already resident in the LPA, and shared routines for the COBOL and PL/I programs.

The VSAM buffers and most of the VSAM file control blocks reside above the 16MB line.

The VSAM buffers may be for CICS data sets defined as using local shared resources (LSR) (that is, the default) or nonshared resources (NSR).

The VSAM LSR pool is built dynamically above the 16MB line when the first file specified as using it is opened, and deleted when the last file using it is closed.

Every opened data set requires some amount of storage in this area for such items as input/output blocks (IOBs) and channel programs.

Files that are defined as data tables use storage above 16MB for records that are included in the table, and for the structures which allow them to be accessed.

QSAM files require some storage in this area. Transient data uses a separate buffer pool above the 16MB line for each type of transient data queue. Storage is obtained from the buffer pool for queue entries as they are added to the destination control table. Transient data also uses a buffer pool below the 16MB line where sections of extrapartition DCTEs are copied for use by QSAM, when an extrapartition queue is being opened or closed.

CICS DBCTL uses DBCTL threads. DBCTL threads are specified in the CICS address space but they have storage requirements in the high private area of the CICS address space.

If DB2 is used by the system, MVS storage is allocated for each DB2 thread.

If you run JVM programs in CICS, that is, run Java programs under control of a Java virtual machine (JVM), CICS uses the MVS JVM which requires significant amounts of MVS storage above and below the 16MB line. For each JVM program running in CICS, there is an MVS JVM running in the CICS address space.

The physical placement of the MVS storage may be anywhere within the region, and may sometimes be above the CICS region. The region may expand into this MVS storage area, above the region, up to the IEALIMIT set by the installation or up to the default value (see the *OS/390 MVS Programming: Callable Services for HLL*). This expansion occurs when operating system GETMAIN requests are issued, the MVS storage within the region is exhausted, and the requests are met from the MVS storage area above the region.

When both the MVS storage areas are exhausted, the GETMAIN request fails, causing abends or a bad return code if it is a conditional request.

The amount of MVS storage must be enough to satisfy the requests for storage during the entire execution of the CICS region. You must use caution here; you never want to run out of MVS Storage but you do not want to overallocate it either.

The size of MVS storage is the storage which remains in the region after allowing for the storage required for the dynamic storage areas, the kernel storage areas, and the IMS/VS and DBRC module storage. The specification of OSCOR storage in CICS/MVS® Version 2 and earlier has been replaced with the specification of the DSA sizes in CICS/ESA Version 3. It is important to specify the correct DSA sizes so that the required amount of MVS storage is available in the region.

Because of the dynamic nature of a CICS system, the demands on MVS storage varies through the day, that is, as the number of tasks increases or data sets are opened and closed. Also, because of this dynamic use of MVS storage, fragmentation occurs, and additional storage must be allocated to compensate for this.

The dynamic storage areas

The dynamic storage areas are used to supply CICS tasks with the storage needed to execute your transactions. From the storage size that you specify on the DSALIM and the EDSALIM parameters, CICS allocates the dynamic storage areas above and below the line respectively.

Too small a dynamic storage area results in increased program compression or, more seriously, SOS (short on storage) conditions, or even storage deadlock abends when program compression is not sufficient.

DSAs consist of one or more extents. An extent below the line is 256KB and above the line, 1MB (except UDSA with TRANISO active, when the extent is 1M).

CICS GETMAIN requests for dynamic storage are satisfied from one of the following: the CDSA, RDSA, SDSA, ESDSA, UDSA, ECDSA, or the ERDSA during normal execution. The sizes of the dynamic storage areas are defined at CICS initialization, but the use of storage within them is very dynamic.

The dynamic storage areas consist of a whole number of virtual storage pages allocated from a number of MVS storage subpools. CICS uses about 180 storage subpools, which are located in the dynamic storage areas. For a list of the subpools see the tables on pages 622 through 630. Each dynamic storage area has it own “storage cushion” These subpools (including the cushion) are dynamically acquired, as needed, a page at a time, from within the dynamic storage area.

The dynamic storage areas are essential for CICS operation. Their requirements depend on many variables, because of the number of subpools. The major contributors to the requirements are program working storage and the type and number of program and terminal definitions. The storage used by individual subpools can be determined by examining the CICS domain subpool statistics (storage manager statistics).

If you have exhausted the tuning possibilities of MVS/ESA and other tuning possibilities outside CICS, and the dynamic storage areas limits have been set as large as possible within CICS, and you are still encountering virtual storage constraint below 16MB, you may have to revise the use of options such as MXT and making programs resident, to keep down the overall storage requirement. This may limit task throughput. If you foresee this problem on an MVS system, you should consider ways of dividing your CICS system, possibly by use of facilities such as CICS multiregion operation (MRO), described in the *CICS Intercommunication Guide*. You can also reduce storage constraint below 16MB by using programs which run above 16MB.

In systems with a moderate proportion of loadable programs, program compression is an indicator of pressure on virtual storage. This can be determined by examining the CICS storage manager statistics which report the number of times that CICS went short on storage.

If the dynamic storage areas limits are too small, CICS performance is degraded. The system may periodically enter a short-on-storage condition, during which it curtails system activity until it can recover enough storage to resume normal operations.

However, resist the temptation to make the dynamic storage area limit as large as possible (which you could do by specifying the maximum allowable region size). If you do this, it can remove any warning of a shortage of virtual storage until the problem becomes intractable.

The dynamic storage areas limits should be as large as possible after due consideration of other areas, especially the MVS storage area above 16MB.

CICS subpools

This section describes briefly the main features of the subpools. They are found in each of the dynamic storage areas. Most of the subpools are placed above the 16MB line. Those subpools that are found below the 16MB line, in the CDSA, SDSA, RDSA, and UDSA, need to be more carefully monitored due to the limited space available. Individual subpools may be static or dynamic. Some contain static CICS storage which cannot be tuned. All the subpools are rounded up to a multiple of 4KB in storage size and this rounding factor should be included in any subpool sizing or evaluation of storage size changes due to tuning or other changes. CICS statistics contain useful information about the size and use of the dynamic storage area subpools. The CICS subpools in the dynamic storage areas may be grouped and described according to the major factor affecting their use.

Application design
: The use of CICS facilities such as program LINK, SHARED storage GETMAINs, the type of file requests, use of temporary storage, application program attributes (resident or dynamic), or the number of concurrent DBCTL, or DB2, requests affect storage requirements.

Number of files definitions
: These subpools may only be tuned by reducing the number of file definitions, or by using MRO.

The use of DBCTL, or DB2
: These subpools are only present if DBCTL or DB2 is used. The subpools may be tuned by reducing the number of threads, or by using maximum tasks (MXT) or transaction classes.

Nucleus and macro table storage
: It may be possible to reduce the macro table storage by reducing the number of macro definitions or by migrating selected macro-defined tables to RDO.

Number and type of terminal definitions
: The OPNDLIM system initialization parameter may also be tuned to limit storage use.

The following tables list the subpools according to their dynamic storage areas and their use.

CICS subpools in the CDSA

Table 191. CICS subpools in the CDSA

Subpool name	Description
DB2	contains the TIE blocks for RMI use, when invoked by the DB2 task-related user exit program, DSN2EXT1. The tie is 100 bytes long and appended to the tie is the local task work area for this task-related user exit which is for DSN2EXT1 2268 bytes long. This subpool is only present when DB2 is used. It may be tuned by limiting DB2 threads or using maximum tasks (MXT) or transaction classes.
DFHAPD24	is a general subpool for application domain. Storage below the line.
DFHTDG24	CXRE queue definitions and SDSCI are allocated from this subpool.
DFHTDSDS	contains real transient data SDSCIs, each of which contains a DCB which resides below the line.
FC_DCB	contains the DCBs for BDAM files. Each file that is defined requires 104 bytes.
FCCBELOW	contains real VSWA and data buffers for pre-reads. Each VSWA requires 120 bytes of storage. The maximum number of data buffers for pre-reads is given by: (number of strings) x (maximum record length) x (number of files).
KESTK24	2KB below the line kernel stack. One per MXT plus one for every dynamic system task that is running.
KESTK24E	4KB below the line kernel stack extension. At least one of these for every ten tasks specified in the MXT limit.
LDNUC	contains the CICS nucleus and macro tables. The CICS nucleus is approximately 192KB and the size of the tables can be calculated. Programs defined EXECKEY(CICS) which are not RESIDENT, that were link edited ˜REENTRANT and RMODE(24).
LDNRS	contains the CICS nucleus and macro tables. The CICS nucleus is approximately 192KB and the size of the tables can be calculated. Programs defined EXECKEY(CICS) and link edited RMODE(24) without the reentrant open.
SMCONTRL	satisfies GETMAINs for control class storage.
SMSHARED	contains shared storage below the 16MB line, for example RMI global work areas, EDF blocks for the life of the transaction being monitored, and other control blocks.
SMSHRC24	is used for many control blocks of SHARED_CICS24 class storage.
SMTCA24	stores the TCA when the task data location option is set to BELOW.
SMTP24	holds line and terminal I/O areas which cannot be located above the 16MB line. The storage requirements depend on the amount of terminal and line traffic in the system. The subpool may be tuned by reducing the RAPOOL, RAMAX, TIOAL size, and number of MRO sessions.
SZSPFCAC	contains the FEPI VTAM ACB work areas.
ZCSETB24	contains application control buffers below the line.

CICS subpools in the SDSA

Table 192. CICS subpools in the SDSA

Subpool name	Description
APECA	contains the event control areas.
DFHAPU24	is a general subpool for application domain storage below the line.
LDPGM	contains dynamically loaded application programs (RMODE (24)). The expected size of this subpool may be predicted from previous releases, and by taking LDPGMRO into account. The subpool size may be reduced by using 31-bit programs. Not reentrant.
LDRES	contains resident application programs (RMODE (24). The expected size of this subpool may be predicted from previous releases, and by taking LDRESRO into account. The subpool size may be reduced by using 31-bit programs. Not reentrant.
OSCOBOL	is used for the allocation of the COBOL merged load list (MLL) control block and its extents. It should never occupy more than its initial allocation of one page of storage.
SMSHRU24	is used for many control blocks of SHARED_USER24 class storage.
ZCTCTUA	contains the TCTTE user area. It can be located in one of the following DSAs: SDSA, ECDSA, CDSA, or ESDSA. Its location is controlled by the system initialization parameter, TCTUALOC=ANY|BELOW and the system initialization parameter, TCTUAKEY=CICS|USER. The maximum size can be specified in USERAREALEN operand of the terminal definition. See the *CICS Resource Definition Guide* for more information about the terminal definition.

CICS subpools in the RDSA

Table 193. CICS subpools in the RDSA

Subpool name	Description
LDNUCRO	contains programs defined EXECKEY(CICS) which are not RESIDENT, that were link edited REENTRANT and RMODE(24).
LDENRS	contains programs defined EXECKEY(CICS) which are RESIDENT, that were link edited REENTRANT and RMODE(24).
LDPGMRO	contains programs defined EXECKEY(USER) which are not RESIDENT, that were link edited RMODE(24) and REENTRANT.
LDRESRO	contains programs defined EXECKEY(USER) and RESIDENT and were link edited REENTRANT and RMODE(24).

CICS subpools in the ECDSA

Table 194. CICS subpools in the ECDSA

Subpool name	Description
AITM_TAE	is the autoinstall terminal model (AITM) table entry subpool (DFHAITDS).
AP_AFCTE	contains the application part of the file control table (FCT) for all local and remote files that are defined. Each VSAM file requires 48 bytes of storage and each remote file requires 64 bytes.
AP_TCA31	contains the application part of the TCA table
AP_TXDEX	contains the application part of the TXD table

Table 194. CICS subpools in the ECDSA (continued)

Subpool name	Description
APBMS	contains storage use by BMS.
APCOMM31	contains COMMAREAs. The storage requirement depends on the size of COMMAREA specified and the number of concurrent users of the application.
APEISTAC	contains storage used by the EXEC CICS interface.
APICAD31	contains storage for ICEs and AIDs above the line.
APLLASYS	contains system load list areas (LLA).
APRSAQCL	contains storage used by the EXEC CICS interface.
APURD	subpool contains URDs and nontask DWEs.
DBCTL	subpool contains the TIE blocks for RMI use, when invoked by the DBCTL task-related user exit program, DFHDBAT. The tie is 120 bytes long, and appended to the tie is the local task work area for this task-related user exit which is, for DFHDBAT, 668 bytes long. This subpool is present only when DBCTL is used. It may be tuned by limiting DBCTL threads or using maximum tasks (MXT) or transaction classes.
DCTE_EXT	contains all extrapartition queue definitions.
DCTE_IND	contains all indirect queue definitions.
DCTE_INT	contains all intrapartition queue definitions.
DCTE_REM	contains all remote queue definitions.
DDBROWSE	contains storage for directory manager browse request tokens.
DDGENRAL	contains directory manager control blocks general information.
DDS_PPT	contains storage for directory manager directory elements for the PPT table.
DDS_RTXD	contains storage for directory manager directory elements for the RTXD table.
DDS_TCL	contains storage for directory manager directory elements for the TCL table.
DDS_TPNM	contains storage for directory manager directory elements for the TPNM table.
DDS_TXD	contains storage for directory manager directory elements for the TXD table.
DDS_USD1	contains storage for directory manager directory elements for the USD1 table.
DDS_USD2	contains storage for directory manager directory elements for the USD2.
DFHAPDAN	is a general subpool for application domain storage above the line.
DFHTDG31	contains transient data general storage and control blocks. The storage requirement depends on the number of buffers and strings, and on the control interval size specified.
DFHTDIOB	contains intrapartition transient data input/output buffers. The storage requirement is given by the control interval size of the intrapartition transient data set multiplied by the number of buffers.
DFHTDWCB	contains the transient data wait elements.
DL/I	subpool contains the TIE blocks for RMI use, when invoked by the EXEC DL/I task-related user exit program, DFHEDP. The tie is 120 bytes long, and appended to the tie is the local task work area for this task-related user exit which is, for DFHEDP, 4 bytes long. This subpool is only present when EXEC DL/I is used. It may be tuned by limiting DBCTL threads or using maximum tasks (MXT) or transaction classes.
DMSUBPOL	is the domain manager subpool for general usage.

Table 194. CICS subpools in the ECDSA (continued)

Subpool name	Description
DSBROWSE	contains storage for dispatcher browse request tokens.
FC_ABOVE	contains real VSWA and data buffers for pre-reads. Each VSWA requires 120 bytes of storage. The maximum number of data buffers for pre-reads is given by: (number of strings) x (maximum record length) x (number of files)
FC_ACB	contains ACBs for VSAM files. There is one ACB, of 80 bytes, per VSAM file.
FC_BDAM	BDAM file control blocks. Each BDAM file requires 96 bytes of storage.
FC_DSNAM	contains data set name blocks. Each file requires a data set name block which uses 120 bytes of storage.
FC_FFLE	contains the fast file elements (FFLEs). A FFLE is built each time a transaction references a file name that has not previously been referenced by that transaction. The FFLE is retained until the end of the task. There is a free chain of FFLEs not currently in use.
FC_FRAB	contains file request anchor blocks (FRABs). There is one FRAB for each transaction that has issued a file control request. The FRAB is retained until the end of the task. There is a free chain of FRABs not currently in use.
FC_FRTE	contains file request thread elements (FRTE). There is one FRTE for each active file control request per task. A file control request has a FRTE if: • It has not yet terminated its VSAM thread. For example, a browse that has not yet issued an ENDBR. • It has updated a recoverable file and there has not yet been a syncpoint. • It is holding READ-SET storage that must be freed in future. There is a free chain of FRTEs not currently in use.
FC_SHRCT	contains file control SHRCTL blocks. There are eight of these and each describes a VSAM LSR pool.
FC_VSAM	contains the file control table (FCT) entries for VSAM files.
FCB_C1K	contains file control buffers of length 1KB. They are used by file control requests that are made against files whose maximum record length is between 512 bytes+1 byte up to 1KB.
FCB_C12K	contains file control buffers of length 12KB. They are used by file control requests that are made against files whose maximum record length is between 8KB+1 byte up to 12KB.
FCB_C16K	contains file control buffers of length 16KB. They are used by file control requests that are made against files whose maximum record length is between 12KB+1 byte up to 16KB.
FCB_C2K	contains file control buffers of length 2KB. They are used by file control requests that are made against files whose maximum record length is between 1KB+ 1 byte up to 2KB.
FCB_C20K	contains file control buffers of length 20KB. They are used by file control requests that are made against files whose maximum record length is between 16KB+1 byte up to 20KB.
FCB_C24K	contains file control buffers of length 24KB. They are used by file control requests that are made against files whose maximum record length is between 20KB+1 byte up to 24KB.
FCB256	contains file control buffers of length 256 bytes. They are used by file control requests that are made against files whose maximum record length is less than or equal to 256 bytes.

Table 194. CICS subpools in the ECDSA (continued)

Subpool name	Description
FCB_C28K	contains file control buffers of length 28KB. They are used by file control requests that are made against files whose maximum record length is between 24KB+1 byte up to 28KB.
FCB_C32K	contains file control buffers of length 32KB. They are used by file control requests that are made against files whose maximum record length is between 28KB+1 byte up to 32KB.
FCB_C4K	contains file control buffers of length 4KB. They are used by file control requests that are made against files whose maximum record length is between 2KB+1 byte up to 4KB.
FCB_C512	contains file control buffers of length 512 bytes. They are used by file control requests that are made against files whose maximum record length is between 256 bytes+1 byte up to 512 bytes.
FCB_C8K	contains file control buffers of length 8KB. They are used by file control requests that are made against files whose maximum record length is between 4KB+1 byte up to 8KB.
KESTK31	24KB above the line kernel stack. One per MXT plus one for every dynamic system task that is running.
KESTK31E	4KB above the line kernel stack extensions. At least one for every ten tasks specified in the MXT limit.
KETASK	kernel task entries.
L2OFL2BL	log manager domain - logger block entries.
L2OFL2BS	log manager domain - logger browseable stream objects.
L2OFL2CH	log manager domain - logger chain objects.
L2OFL2SR	log manager domain - logger stream objects.
LD_APES	loader domain - active program elements.
LD_CNTRL	loader domain - general control information.
LD_CSECT	loader domain - CSECT list storage.
LDENUC	contains the extended CICS nucleus, abd 31–bit macro tables. The extended CICS nucleus is approximately 50KB. Programs defined EXECKEY(CICS) which are not RESIDENT, that were link edited REENTRANT and RMODE (24).
LDNRSRO	Contains programs defined EXECKEY(CICS) which are RESIDENT, that were link edited REENTRANT and RMODE (24).
LGBD	log manager domain - log stream name/journal name/journal model browse tokens.
LGGD	log manager domain - explicitly opened general logs.
LGGENRAL	general purpose subpool for log manager domain.
LGJI	log manager domain - journal name entries.
LGJMC	log manager domain - journal model resource entries.
LGSD	log manager domain - log stream data entries.
LGUOW	log manager domain - unit of work data entries.
LI_PLB	language interface - program language block. One is allocated for each program when control is first passed to it.
MDTTABLE	MDT field attribute table for BMS maps sent through the CICS Web interface.
MN_CNTRL	contains monitoring control blocks - general information.

Table 194. CICS subpools in the ECDSA (continued)

Subpool name	Description
MN_MAES	contains monitoring control blocks. The storage requirement is 48 bytes per active task.
MN_TMAS	contains monitoring control blocks. The storage requirement is 472 bytes per active task.
MRO_QUEU	is used by the MRO work queue manager.
MROWORKE	is used by the MRO work queue manager elements.
PGGENRAL	general purpose program manager domain subpools.
PGHMRSA	program handle manager cobol register save areas.
PGHTB	program manager handle table block.
PGLLE	program manager load list elements.
PGPGWE	program manager wait elements.
PGPPTE	program manager program definitions (PPTs).
PGPTA	program manager transaction-related information.
PR_TABLE	contains storage for PTEs from the PRT.
RM_TABLE	is used to quickcall the recovery manager lifetime control block.
SMSHRC31	is used for many control blocks of SHARED_CICS31 class storage.
SMTCA	stores the TCA when the task data location option is set to ANY.
SMTP	holds line and terminal I/O areas. The storage requirements depend on the amount of terminal and line traffic in the system. The subpool may be tuned by reducing the RAPOOL, RAMAX, TIOAL size, and number of MRO sessions.
STSUBPOL	is a statistics domain manager subpool.
SZSPFCCD	is the FEPI connection control subpool.
SZSPFCCM	is the FEPI common area subpool.
SZSPFCCV	is the FEPI conversation control subpool.
SZSPFCDS	is the FEPI device support subpool.
SZSPFCNB	is the FEPI node initialization block subpool.
SZSPFCND	is the FEPI node definition subpool.
SZSPFCPD	is the FEPI pool descriptor subpool.
SZSPFCPS	is the FEPI property descriptor subpool.
SZSPFCRP	is the FEPI request parameter list subpool.
SZSPFCRQ	is the FEPI requests subpool.
SZSPFCSR	is the FEPI surrogate subpool.
SZSPFCTD	is the FEPI target descriptor subpool.
SZSPFCWE	is the FEPI work element subpool.
SZSPVUDA	is the FEPI data areas subpool.
TD_TDCUB	contains all the transient data CI update control blocks.
TD_TDQUB	contains all the transient data queue update control blocks.
TD_TDUA	contains all the transient data UOW anchor control blocks.
TIA_POOL	is the timer domain anchor subpool.
TIQCPOOL	is the timer domain quickcell subpool.

Table 194. CICS subpools in the ECDSA (continued)

Subpool name	Description
TSBUFFRS	contains the temporary storage I/O buffers. The storage requirement is given by: (TS control interval size) x (number of TS buffers). The use of temporary storage by application programs affects the size of a number of subpools associated with temporary storage control blocks:
TSGENRAL	The amount of storage used by the TSGENRAL subpool depends on the number of buffers and strings and the control interval size defined for the temporary storage data set.
TSMAIN	contains storage for temporary storage main storage. The subpool could be reduced by using auxiliary temporary storage.
TSMN0064	contains TS main items with lengths (including the header) less than or equal to 64.
TSMN0128	contains TS main items with lengths (including the header) less than or equal to 128.
TSMN0192	contains TS main items with lengths (including the header) less than or equal to 192.
TSMN0256	contains TS main items with lengths (including the header) less than or equal to 256.
TSMN0320	contains TS main items with lengths (including the header) less than or equal to 320.
TSMN0384	contains TS main items with lengths (including the header) less than or equal to 384.
TSMN0448	contains TS main items with lengths (including the header) less than or equal to 448.
TSMN0512	contains TS main items with lengths (including the header) less than or equal to 512.
TSTSS	contains TS section descriptors.
TSTSX	contains TS auxiliary item descriptors.
TSDTN	contains TS digital tree nodes.
TSQUEUE	contains TS queue descriptors.
TSBRB	contains TS browse blocks.
TSQAB	contains TS queue anchor blocks.
TSQOB	contains TS queue ownership blocks.
TSTSI	contains TS item descriptors.
TSQUB	contains TS queue update blocks.
TSICDATA	contains TS interval control elements.
TSW	contains TS wait queue elements.
UE_EPBPL	is the subpool for the user exit program block (EPB).
USGENRAL	is the general-purpose subpool for the user domain.
USIDTBL	contains the attach security userid table entries (LUITs). See “ISC/IRC attach time entries” on page 404 for more information.
USRTMQUE	contains queue elements for users waiting for USRDELAY. Each queue element is 16 bytes.

Table 194. CICS subpools in the ECDSA (continued)

Subpool name	Description
USUDB	contains user data blocks. The storage requirement is 128 bytes per unique user.
USXDPOOL	contains user domain transaction-related data. Each executing transaction requires 32 bytes.
VCTRLSUB	is used by volume control's series definition table mechanism.
XMGENRAL	is the general-purpose subpool for the transaction manager
XMLQEA	contains storage for large QEAs. The storage requirement depends on the number of concurrent ENQs for resources.
XMSQEA	contains storage for small QEAs. The storage requirement depends on the number of concurrent ENQs for resources.
XMTCLASS	contains the transaction manager tranclass definition.
XMTRANSN	transaction manager transactions. One for every transaction in the system.
XMTXDINS	transaction manager transaction definition.
XMTXDSTA	transaction manager transaction definition.
XMTXDTPN	contains the transaction manager transaction definition TPNAME storage.
XSGENRAL	is the general-purpose subpool for the security domain.
XSXMPOOL	contains security domain transaction-related data. Each executing transaction requires 56 bytes.
ZCBIMG	contains BIND images.
ZCBMSEXT	contains the BMS extensions for terminals. Subpool storage requirements are 48 bytes for each terminal, surrogate, ISC session, and console.
ZCBUF	contains the non-LU6.2 buffer list.
ZCCCE	contains the console control elements. Each console requires 48 bytes.
ZCGENERL	is the general-purpose subpool for terminal control.
ZCLUCBUF	contains the LU6.2 SEND and RECEIVE buffer list .
ZCLUCEXT	contains the LU6.2 extensions. The storage requirement is 224 bytes for each LU6.2 session.
ZCNIBD	contains the NIB descriptors. Each terminal, surrogate, ISC session, and system definition requires 96 bytes of storage.
ZCNIBISC	contains the expanded NIB and response during OPNDST/CLSDST for ISC. Each concurrent logon/logoff requires 448 bytes of storage. The maximum number of concurrent requests is limited by the number of sessions. The storage may be tuned by reducing the number of sessions.
ZCNIBTRM	contains the expanded NIB during OPNDST/CLSDST for terminals. Each concurrent logon/logoff requires 192 bytes of storage. The maximum number of concurrent requests is limited by the number of terminals . The storage may be tuned by reducing the number of terminals.
ZCRAIA	contains the RECEIVE ANY I/O areas.
ZCRPL	contains the RPLs for active tasks. Each active task associated with a VTAM terminal requires 152 bytes.
ZCSETB	contains application control buffers above the line.
ZCSKEL	contains the remote terminal entries. Each remote terminal definition requires 32 bytes of storage.

Table 194. CICS subpools in the ECDSA (continued)

Subpool name	Description
ZCSNEX	contain the TCTTE sign-on extensions. The storage requirement is 48 bytes for each terminal, surrogate, session, and console.
ZCTCME	contains the mode entries. Each mode entry requires 128 bytes of storage.
ZCTCSE	contains the system entries. Each system entry requires 192 bytes of storage.
ZCTCTTEL	contains the large terminal entries. 504 bytes of storage are required for every terminal, surrogate model, and ISC session defined.
ZCTCTTEM	contains the medium terminal entries. 400 bytes of storage are required for every IRC batch terminal.
ZCTCTTES	contains the small terminal entries. 368 bytes of storage are required for every MRO session and console.
ZCTPEXT	the TPE extension.
ZC2RPL	contains the duplicate RPLs for active tasks. Each active task associated with a VTAM terminal requires 304 bytes.
ZCTCTUA	contains the TCTTE user area. It can be located in one of the following DSAs: CSDSA, SDSA, ECDSA, or ESDSA. Its location is controlled by the system initialization parameter, TCTUALOC=ANY|BELOW and the system initialization parameter, TCTUAKEY=CICS|USER. The maximum size can be specified in USERAREALEN operand of the terminal definition. See the *CICS Resource Definition Guide* for more information.

CICS subpools in the ESDSA

Table 195. CICS subpools in the ESDSA

Subpool name	Description
LDEPGM	contains extended (31) bit dynamically loaded application programs and programs defined EXECKEY(USER).
LDERES	contains extended (31) bit resident application programs.
SMSHRU31	is used for many control blocks of SHARED_USER31 class storage, RMI global work areas, EDF blocks for the life of the transaction being monitored, and other control blocks.
ZCTCTUA	contains the TCTTE user area. It can be located in one of the following DSAs: CDSA, SDSA, ECDSA, or ESDSA. Its location is controlled by the system initialization parameter, TCTUALOC=ANY|BELOW and the system initialization parameter, TCTUAKEY=CICS|USER. The maximum size can be specified in USERAREALEN operand of the terminal definition. See the *CICS Resource Definition Guide* for more information.

CICS subpools in the ERDSA

Table 196. CICS subpools in the ERDSA

Subpool name	Description
LDENUCRO	Contains the extended CICS nucleus and 31–bit macro tables. The extended CICS nucleus is approximately 1850KB. Programs defined EXECKEY(CICS) which are not RESIDENT, that were link edited REENTRANT and RMODE(24).

Table 196. CICS subpools in the ERDSA (continued)

Subpool name	Description
LDENRSRO	Contains the extended CICS nucleus and 31–bit macro tables. The extended CICS nucleus is approximately 1850K. Programs defined EXECKEY(CICS) which are RESIDENT, that were link edited REENTRANT and RMODE(24)
LDEPGMRO	contains extended (31) bit dynamically loaded application programs. The contents of this subpool has to linked reentrant.
LDERESRO	contains extended (31) bit resident application programs. The contents of this subpool has to linked reentrant.

Short-on-storage conditions caused by subpool storage fragmentation

When migrating to CICS Transaction Server for OS/390 you may experience short-on-storage problems in the Dynamic Storage Areas (DSAs) below the 16M line. In many cases the amount of storage allocated to the region is greater than in previous releases.

CICS storage management incorporates transaction isolation (subspaces) support in CICS Transaction Server for OS/390, including the fact that the DSAs are now managed by CICS with the DSA/EDSA limits being specified in the SIT, rather than a size for each DSA as in previous releases. Storage extents support dynamic storage management and provide subspace support. Storage extents are always allocated in multiples of 256K below the 16M line, with the exception of the UDSA which is allocated in 1M extents when transaction isolation is in use. Above the line extents are allocated in multiples of 1M.

When a DSA, such as the CDSA, requires additional storage in order to satisfy a GETMAIN request, the CICS storage manager allocates another extent to that DSA. However, if all extents are currently allocated, an attempt is made to locate a free extent in another DSA which may then be relocated to the DSA in need. However, in order to remove an extent from one DSA so that it may be allocated to another, all pages in the extent must be free (that is, not allocated to any subpool).

Analysis of short-on-storage problems begins by obtaining a dump when the system is in a short-on-storage condition. The best documentation is obtained by setting an entry in the dump table causing a dump to be taken when the DFHSM0131 (short-on-storage below the line) or DFHSM0133 (short-on-storage above the line) is issued. Use the CICS command CEMT SET SYDUMPCODE(SM0131) SYSDUMP MAXIMUM(1) ADD to indicate that a dump should be taken the first time a DFHSM0131 message is issued.

Use the IPCS command VERBX CICS530 'SM=3' to format the SM control blocks. Examine the DSA summaries, noting which DSA(s) are short-on-storage and the amount of free space in the other DSAs (above or below the 16M line as appropriate). The amount of freespace is given for each extent for each DSA.

Frequently either the UDSA or the CDSA is short-on-storage but there is a large amount of free storage in the SDSA. The following dump extracts are from a problem of this type where the UDSA is short-on-storage.

Each extent has an associated page pool extent (PPX) and page allocation map (PAM). Examination of the SDSA extents shows several extents with large amounts

of freespace. For example, the extent beginning at 00700000 running through 0073FFFF has only 4K allocated and 252K free.

```
Extent list:      Start      End          Size     Free
                  00700000   0073FFFF     256K     252K
```

The DSA extent summary shows that the PPX for the extent at 00700000 is found at 09F0A100, and the associated PAM is found at 09F0A150. Examination of the PAM shows only one page is allocated, and it belongs to the subpool with an ID of x'7A'.

```
  Start      End          Size   PPX_addr   Acc     DSA
  00700000   0073FFFF     256K   09F0A100    C     SDSA

  PPX.SDSA 09F0A100 Pagepool Extent Control Area

     0000  00506EC4 C6C8E2D4 D7D7E740 40404040  *.&>DFHSMPPX     *
     0010  E2C4E2C1 40404040 09A1BA68 071B3EA0  *SDSA    ........*
     0020  00040000 00700000 0073FFFF 071B5EE0  *.............. *
     0030  00000000 09F0A150 00000040 0710A268  *.....0.&;.. ..s.*
     0040  0003F000 00000000 00000000 00000000  *..0.............*

  PAM.SDSA 09F0A150 Page Allocation Map

     0000  00000000 00000000 00000000 00000000  *................*
     0010 -    002F LINES SAME AS ABOVE
     0030  00000000 0000007A 00000000 00000000  *................*
```

The domain subpool summary determines for the SDSA which subpool is associated with the ID of x'7A'. In this dump 7A is the ID for subpool ZCTCTUA. Do not rely on the IDs being the same for multiple runs of CICS because the IDs are assigned in the order the ADD_SUBPOOL is issued.

```
  ==SM: UDSA Summary (first part only)

     Size:                            512K
     Cushion size:                     64K
     Current free space:               56K  (10%)
   * Lwm free space:                   12K  ( 2%)
   * Hwm free space:                  276K  (53%)
     Largest free area:                56K
   * Times nostg returned:              0
   * Times request suspended:           0
     Current suspended:                 0
   * Hwm suspended:                     0
   * Times cushion released:            1
     Currently SOS:                   YES

  ==SM: SDSA Summary (first part only)

     Size:                           4352K
     Cushion size:                     64K
     Current free space:             2396K  (55%)
   * Lwm free space:                  760K  (17%)
   * Hwm free space:                 2396K  (55%)
     Largest free area:               252K
   * Times nostg returned:              0
   * Times request suspended:           0
     Current suspended:                 0
   * Hwm suspended:                     0
   * Times cushion released:            0
     Currently SOS:                    NO
```

A short-on-storage condition can occur as a result of large amounts of redundant program storage (RPS). This can be identified in the domain subpool summary and the loader domain summary (use the IPCS command VERBX CICS 530 'LD=3' to format the LD control blocks).

DFH0STAT provides useful information in the storage summary without a breakdown by subpool. DFH0STAT should be run just prior to the completion of the statistics interval. For example, if the statistics interval is 3 hours, run DFH0STAT at 2 hours and 59 minutes. See "Appendix E. The sample statistics program, DFH0STAT" on page 513 for more information.

To ease the short-on-storage problems, you can add records to the local catalog to enable the CICS self-tuning mechanism for storage manager domain subpools. For information about how to do this, using the CICS-supplied utility program, DFHSMUTL, see *CICS Operations and Utilities Guide*. Alternatively, you can fix the size of the DSA using one or more of the SIT overrides, CDSASZE, UDSASZE, SDSASZE, RDSASZE, ECDSASZE, EUDSASZE, ESDSASZE, and ERDSASZE, (see the *CICS System Definition Guide*). The self-tuning mechanism and the SIT overrides should only be used if increasing the DSA or EDSA limits does not completely resolve the short-on-storage problems.

Storage management requests the loader to reduce the RPS storage below 80%. This makes additional extents available to be allocated to the DSA in need.

LDPGMRO storage is allocated on a 16-byte boundary to reduce free space between programs.

If short-on-storage problems persist, initial DSA sizes may be specified as SIT overrides. The following process can be used to determine the values to use.

Collect DFH0STAT output as described, for information showing storage use by DSA during the intervals.

Review the CICS statistics for several days. This provides information which can be used to define the amount of storage used at a subpool and a DSA level. Extent usage is shown with the number of extents added and released.

In addition to the DSA information provided in DFH0STAT, the results about each subpool are provided, including the DSA where it was allocated. If statistics are being gathered, end-of-day statistics will only provide data since the last statistics collection.

Determine if DSALIM has been specified as large as possible, but allowing for OSCORE requirements of the various packages in use.

Allocating into managed extents can lead to a block of storage in an extent which is insufficient to satisfy a getmain request. With the dynamic nature of the subpools and DSAs, this should be relieved as the subpool/extent storage is reused. Specifying the initial DSA size using the SIT override for the affected DSA has the effect of reserving contiguous extents up to the amount specified, and eliminating the blocks of storage.

Additional DSAs (RDSA and SDSA) are available and many of the subpools from the UDSA are moved to the SDSA. The end-of-day statistics or information in a dump of the CICS region can be used to define relative sizes of the subpools and associated DSAs.

Also, using the LPA reduces the amount of storage used in LDNUCRO by approximately 100K.

MAPS should be defined as MAPS. Defining MAPS as programs causes them to be loaded into LDRES rather than in LDNUC. LDRES is part of the SDSA and more sensitive to fragmentation. For PSBPOOL space, the shutdown statistics provide the correct size.

CICS kernel storage

CICS kernel storage consists of control blocks and data areas that CICS requires to manage system and user tasks throughout CICS execution. The majority of this storage is allocated from the CICS DSAs. A small amount of this storage is allocated from MVS storage.

The kernel recognises two types of task: static tasks, and dynamic tasks. The kernel storage for static tasks is pre-allocated and is used for tasks controlled by the MXT mechanism. The storage for dynamic tasks is not pre-allocated and is used for tasks such as system tasks which are not controlled by the MXT value. Because the storage for dynamic tasks is not pre-allocated, the kernel may need to GETMAIN the storage required to attach a dynamic task when the task is attached.

The number of static tasks is dependant upon the current MXT value (there are MXT+1 static tasks). The storage for static tasks is always GETMAINed from the CICS DSAs. If MXT is lowered the storage for an excess number of static tasks is freed again.

During early CICS initialization the kernel allocates storage for 8 dynamic tasks. This storage is GETMAINed from MVS and is always available for use by internal CICS tasks. All other storage for dynamic tasks is then allocated, as needed, from the CICS DSAs. Typically when a dynamic task ends, its associated storage is freed.

The storage required by a single task is the same for both types of task and can be divided into storage required above and below the 16MB line:

- Above the line the following storage is required per task:
  - A 896-byte kernel task entry
  - A 24K 31-bit stack.
- Below the line the following storage is required per task:
  - A 2K 24-bit stack.

In addition to this storage, the kernel also allocates a number of 4K extension stacks both above and below the 16MB line. These are for use by any task, if it overflows the stack storage allocated to it. The number of 24-bit and 31-bit stack extensions pre-allocated by the kernel is determined by dividing the current MXT value by 10.

When the kernel GETMAINs storage from the CICS DSAs, the following subpools are used:

- In the CDSA:

```
KESTK24             2K stack segments
KESTK24E            4K extension stack segments
```

- In the ECDSA:

```
KESTK31             24K stack segments
KESTK31E            4K extension stack segments
KETASK              896 byte task entries
```

Appendix G. Performance data

This appendix contains the relative costs of a subset of the CICS application program interface (API) calls. The information is divided into the following sections:

Using the tables in this appendix, you can compare the relative processing times of particular CICS API calls, and examine some of the other factors that affect overall processing times. These tables can help you make decisions concerning application design when you are considering performance. To calculate a time for a transaction, find the entries appropriate to your installation and application, and add their values together.

Before you work with these numbers, please note the following:

- The cost per call is documented in 1K or millisecond instruction counts taken from a tracing tool used internally by IBM. Each execution of an instruction has a count of 1. No weighting factor is added for instructions that use more machine cycles than others.
- Because the measurement consists of tracing a single transaction within the CICS region, any wait for I/O etc. results in a full MVS WAIT. This cost has been included in the numbers reported in this document. On a busy system the possibility of taking a full MVS WAIT is reduced because the dispatcher has a higher chance of finding more work to do.
- When judging performance, the numbers in this book should not be compared with those published previously, because a different methodology has been used.

Variable costs

The sections from "Transaction initialization and termination" on page 640 onwards describe the relative costs of a subset of the CICS API calls. To those costs must be added the variable costs described in this section.

Variable costs are encountered, for different machine configurations, when there is synchronous access to a coupling facility. For example, RLS and shared temporary storage use synchronous access to a coupling facility; so, for CF log streams, does the MVS logger. The variance occurs because a synchronous access instruction executes for as long as it takes to complete the access to the coupling facility and return. The number of central processing unit (CPU) cycles consumed during the request therefore depends on:

- The speed of access to the coupling facility.

- The speed of the processor CPU. Assuming that the access time to a particular coupling facility is a constant, if the CPU speed were to be changed the number of CPU cycles consumed during the request would also change.

The following sections describe variable costs for logging and syncpointing.

Logging

Because logging costs contain some of the variable costs incurred by synchronous accesses to the coupling facility, they are documented here in terms of milliseconds of CPU time. The measurements have been taken on a 9672-R61 with a 9674-R61 coupling facility; they can be scaled to any target system, using the IT Relative Ratios (ITRRs) published in the *IBM Large System Performance Report*. This can be accessed through the IBM System/390 web page (http://www.s390.ibm.com), more specifically, at *http://www.s390.ibm.com/lspr/lspr.html.*

When looking at the cost of accessing recoverable resources, the cost of writing the log buffer to primary storage has been separated from the API cost. FORCE and NOFORCE are the two types of write operations to the system log buffer.

- The FORCE operation requests that the log buffer is written out and is made non-volatile. The transaction that made this request is suspended until the process completes. The log is not written out immediately but is deferred using an internal algorithm. The first forced write to the log sets the clock ticking for the deferred log flush. Subsequent transactions requesting log forces will put their data in the buffer and suspend until the original deferred time has expired. This permits buffering of log requests and it means that the cost of writing the log buffer is shared between many transactions.
- The NOFORCE operation puts the data into the log buffer, which is written to primary storage when a FORCE operation is requested or the buffer becomes full.

The cost of writing a log buffer varies, depending on which of the following applies:

- The write is synchronous to the coupling facility
- The write is asynchronous to the coupling facility
- A staging data set is being used
- DASD-only logging is being used.

Synchronous writes to the CF

Writes of less than 4K in size are generally synchronous. A synchronous write uses a special instruction that accesses the coupling facility directly. The instruction lasts for as long as it takes to access the coupling facility and return. This access time, known as the "CF Service Time", depends on both the speed of the coupling facility and the speed of the link to it. CF Service Times can be monitored using RMF III, as shown on page 271. For synchronous writes, the CPU cost of the access changes as the CF Service Time changes; this is not true of asynchronous writes.

Asynchronous writes to the CF

Asynchronous writes do not use the same instruction used by synchronous writes. A CICS task that does an asynchronous log write gives up control to another task, and the operation is completed by the logger address space.

Table 197 shows the costs of the various flavours of log writes. Note that, although CICS Transaction Server for OS/390 log writes are more expensive than those in pre-CICS Transaction Server for OS/390 releases, a change in the logging algorithm means that the frequency of logging is less.

Table 197. Costs of log writes. The measurements were taken on a 9672-R61 with a 9674-R61 coupling facility.

Type of log write	CPU time in milliseconds	
	CICS address space	**MVS logger address space**
CF synchronous (i.e. < 4K)	1.754 ★	-
CF asynchronous (i.e. > 4K)	2.354	0.771
Staging data set < 4K	2.805	0.881
Staging data set > 4K	1.939	1.520
DASD-only < 4K	2.678	0.703
DASD-only > 4K	2.680	0.720

★: For a synchronous access to the coupling facility, the figure for the CPU time in the CICS address space includes the CF Service Time of 0.451ms. Note that, for a synchronous write to the coupling facility:

- The CPU time in the CICS address space is affected by the speed of the coupling facility.
- The *proportion* of the CPU time in the CICS address space represented by the CF Service Time will vary, depending on the CPU and coupling facility used.
- This measurement was taken using a buffer size of 3800 bytes. Smaller buffer writes use less CF service time.

Syncpointing

The syncpoint cost needs to be factored into the overall transaction cost. The amount of work at syncpoint varies according to the number of different types of resource managers (RMs) involved during the unit of work (UOW). Therefore, the cost can vary.

Typically, a syncpoint calls all the RMs that have been involved during the UOW. These may or may not need to place data in the log buffer before it is written out. For example, recoverable TD defers putting data into the log buffer until a syncpoint. Recovery manager itself puts commit records into the log buffer and requests a forced write. For these reasons it is difficult to give a precise cost for a syncpoint, but the following should be used as a guide:

A syncpoint can be split as follows:

Basic cost	5.0
Put commit records in the log buffer	2.0
For each RM used in UOW	2.5
Write log buffer	See “Logging” on page 638

This shows syncpoint costs, in 1K instruction units, for local resources only. If distributed resources are updated, communication costs will need to be added.

If no recoverable resources have been updated, the cost is only the transaction termination cost as described in “Transaction initialization and termination” on page 640.

Additional costs

The calculations in the following sections have been made assuming that performance monitoring and CICS tracing are turned off. Monitoring and tracing incur additional costs.

Using an internal IBM benchmark with a pathlength of 20 milliseconds of CPU time on a 9672-R61, trace added about 20% to the transaction pathlength. Performance monitoring added about 5% to the transaction pathlength.

Transaction initialization and termination

This section shows the costs for the following:

- Receive
- Attach/terminate
- Send.

Receive

The receive cost is based on an LU2 type terminal sending a 4-byte transaction identifier and includes all the VTAM processing using HPO=YES.

Receive transaction ID	13.5

Attach/terminate

	Assembler	COBOL
Attach and initialization	7.5	11.0
Termination	6.2	10.0

Notes:

The transaction initialization cost is calculated from the start of transaction attach to the start of the CICS application code.

The transaction termination cost assumes that no recoverable resources have been updated. If recoverable resources have been updated, the syncpointing cost must be added to the termination cost.

Send

The send cost consists of one request unit to a LU2 type terminal. It includes both CICS and VTAM instructions for a system using HPO=YES.

Send to terminal	17.0

File control

This section contains the relative costs of VSAM file control accesses. For read operations the VSAM I/O cost is not included because the necessity to access DASD is workload dependent. For the read operation to complete both the index and data must be accessed. If neither index or data is in a buffer, an I/O must be

done for each level of index and one for the data. The relative number of instructions, in 1K instruction counts, for the I/O for each file type is as follows:

- 9.5 for a KSDS
- 9.5 for an ESDS
- 8.2 for a RRDS

READ

KSDS	ESDS	RRDS	Data Table (CMT)
3.0	2.4	2.2	First: 1.5 Subsequent:1.1

READ UPDATE

Recoverable and non-recoverable files are included in the READ UPDATE cost:

Non-recoverable files

KSDS	ESDS	RRDS
3.1	2.3	2.2

Recoverable files

KSDS	ESDS	RRDS
5.5	4.3	4.2
Notes: A recoverable READ UPDATE puts the 'before image' into the log buffer which, if not subsequently written to primary storage, is written out before the REWRITE is completed.		

REWRITE

Recoverable and non-recoverable files are included in the REWRITE cost.

Every REWRITE has a data VSAM I/O associated with it.

Non-recoverable files

KSDS	ESDS	RRDS
10.2	10.1	10.1

Recoverable files

KSDS	ESDS	RRDS
10.4	10.3	10.3

KSDS	ESDS	RRDS
Notes: A REWRITE of a recoverable file requires that the log buffer containing the *before image* has been written out. If the buffer has not already been written out since the READ UPDATE, the cost of writing the log buffer is incurred. When the before image has been hardened the VSAM I/O takes place. At the end of the transaction, there are additional costs involved in syncpointing if recoverable resources have been updated. See "Syncpointing" on page 639.		

WRITE

The cost for WRITE includes nonrecoverable files and recoverable files.

Every WRITE has a data VSAM I/O associated with it. The index will need to be written only when a control area split occurs.

Non-Recoverable files

KSDS	ESDS	RRDS
12.9	11.1	10.9

Recoverable files

KSDS	ESDS	RRDS
14.9	13.1	12.9
Notes: Every WRITE has a hidden READ associated with it to ensure that the record is not already present in the file. This under the cover READ could incur the cost of I/Os if the index and/or data are not in the buffer. Each WRITE to a recoverable file will require that the Log Buffer containing the data image has been written out before doing the VSAM I/O. At the end of the transaction, there are additional costs involved in syncpointing if recoverable resources have been updated. See "Syncpointing" on page 639.		

DELETE

You cannot delete from an ESDS record file.

Non-Recoverable files

KSDS	RRDS
12.5	11.5

Recoverable files

KSDS	RRDS
14.5	13.5

KSDS	RRDS
Notes: At the end of the transaction, additional costs are involved in syncpointing if recoverable resources have been updated. See "Syncpointing" on page 639.	

Browsing

STARTBR	READNEXT	READPREV	RESETBR	ENDBR
3.1	1.5	1.6	2.6	1.4

UNLOCK

The pathlength for EXEC CICS UNLOCK is 0.7.

Coupling facility data tables

The CPU instruction data provided here was obtained using a 9672-R55 system.

Two tables are provided:

- The first for record lengths that result in synchronous coupling facility accesses (less than 4K)
- The second for record lengths that result in asynchronous coupling facility accesses (greater than 4K).

Note that the asynchronous requests do take more CPU time to process. The response times will also be slightly longer than for synchronous requests. CPU instructions per API call for record lengths less than 4K are as follows:

API CALL	CONTENTION	LOCKING	RECOVERABLE
READ	11.8	11.8	11.8
READ/UPDATE	12.0	22.2	22.4
REWRITE	19.5	24.0	33.0
WRITE	8.0	8.0	13.0
DELETE	7.0	11.0	16.5

CPU instructions per API call for record lengths greater than 4K, are

API CALL	CONTENTION	LOCKING	RECOVERABLE
READ	15.3	15.3	15.3
READ/UPDATE	15.0	25.7	25.9
REWRITE	23.0	27.5	36.5
WRITE	11.5	11.5	16.5
DELETE	10.5	14.5	20.0

Record Level Sharing (RLS)

For information about performance measurements on record level sharing (RLS), see the *System/390 MVS Parallel Sysplex Performance* manual, SG24 4356 02.

Temporary Storage

The costing for temporary storage covers the following:

- Main storage

In each example, *n* represents the number of items in the queue before it is deleted.

Main Storage

WRITEQ	REWRITE	READQ	DELETEQ
1.0	0.8	0.8	0.71 + 0.23 * n

Auxiliary Storage

The approximations for auxiliary TS queues do not include any VSAM I/O cost. A VSAM I/O costs approximately 11.5K instructions and will occur as follows:

- When attempting to write an item that does not fit in any buffer
- When reading an item that is not in the buffer
- If, when reading a control interval from DASD with no available buffer space, the least recently used buffer must first be written out.

Therefore, under certain circumstances, a READQ could incur the cost of two VSAM I/Os.

Non-Recoverable TS Queue

WRITEQ	REWRITE	READQ	DELETEQ
1.3	1.8	1.0	0.75 + 0.18 * n

Recoverable TS Queue

WRITEQ	REWRITE	READQ	DELETEQ
1.4	19	1.0	0.87 + 0.18 * n

Note: The main difference between the cost of accessing non-recoverable and recoverable TS queues is incurred at syncpoint time, when, for recoverable queues, the following happens:

- The VSAM I/O cost is incurred if only interval has been used during the unit of work, and has not already reached DASD.
- The new DASD control interval addresses are put in the log buffer. The cost for recovery manager to do this is about 2.0K instructions.
- A forced log write is requested and the syncpoint will complete when the log buffer has been written to primary storage. For more information, see "Variable costs" on page 637.

Shared Temporary Storage

For information about performance measurements on shared temporary storage, see the *System/390 MVS Parallel Sysplex Performance* manual.

Transient Data

Transient data costs in this section are for the following:

- Intrapartition Queues
- "Extrapartition queues"

Intrapartition Queues

The approximations for non-recoverable and logically recoverable intrapartition TD queues do not include any VSAM I/O cost. A **VSAM I/O costs approximately 11.5K** and occurs:

- When attempting to write an item that will not fit in any buffer.
- When reading an item that is not in the buffer.
- If, when reading a control interval from DASD and there is no available buffer space, the least recently used buffer will first have to be written out. Therefore, under certain circumstances, a READQ could incur the cost of two VSAM I/Os.

Non-Recoverable TD Queue

WRITEQ	READQ	DELETEQ
1.5	1.3	1.3

Logically Recoverable TD Queue

WRITEQ	READQ	DELETEQ
First: 2.8 Subsequent:1.5	First: 2.4 Subsequent:1.4	1.1

Notes:

The main difference between Non-Recoverable and Logically Recoverable TD Queues occurs at Syncpoint time. At syncpoint, the new TD Queue addresses are put in the Log Buffer and a forced Log write is requested. The cost to put the data in the buffer is about 2.0K. The cost of writing the Log Buffer to the CF is described in the section on Recovery Costs.

Physically Recoverable TD Queue

WRITEQ	READQ	DELETEQ
19.7	First: 9.3 Subsequent:8.8	8.7
Notes: Physically Recoverable WRITEQ requests involve forcing a VSAM I/O and forcing a Log write to the CF for every request.		

Extrapartition queues

The approximate calculations for extrapartition TD queues do not include any I/O cost. An **I/O for a physically sequential file costs approximately 7.0K** and occurs as follows:

- When attempting to write an item that will not fit in any buffer.
- When reading an item that is not in the buffer.

- If, when reading data from DASD and there is no available buffer space, the least recently used buffer will first have to be written out.

Therefore, under certain circumstances, a READQ could incur the cost of two I/Os.

Extrapartition TD queues are non-recoverable.

WRITEQ	READQ
1.2	1.0

Program Control

Program control costs assume that all programs have previously been loaded, and that there is no load operation from DASD.

	Assembler	COBOL
LINK	1.5	4.0
XCTL	2.1	5.1
RETURN	1.1	3.3

Storage control

GETMAIN	FREEMAIN
0.9	0.9

Interregion Communication

This section describes the additional costs of communication between two CICS regions using the following communication methods:

MRO XM
This is CICS to CICS communication where both regions are in the same MVS image. CICS uses MVS cross memory (XM) services for this environment.

MRO XCF (via CTC)
This is CICS to CICS communication where both regions are on separate MVS images. In this environment the transport class is defined to use a XCF path that exploits a channel to channel (CTC) device for message traffic between the two MVS images. This is only supported within a sysplex.

MRO XCF (via CF)
This is CICS to CICS communication where both regions are on separate MVS images. In this environment the transport class is defined to use an XCF path that exploits a CF structure for message traffic between the two MVS images. This is supported only within a sysplex.

ISC LU6.2
This is CICS to CICS communication where both regions are on separate MVS images. In this environment VTAM LU6.2 uses a CTC for communication between the two MVS images.

Transaction routing

MRO XM	MRO XCF (via CTC)	MRO XCF (via CF)	ISC LU6.2
37.0	43.0	66.0	110.0

Function shipping (MROLRM=YES)

	MRO XM	MRO XCF (via CTC)	MRO XCF (via CF)
Initiate/terminate environment	13.2	13.2	13.2
Each function shipping request	9.0	23.4	48.4
Syncpoint flow	9.0	23.4	48.4

Notes:

The above costs relate to CICS systems with long running mirrors.

ISC LU6.2 does not support MROLRM=YES.

Included in the initiate/terminate environment is the cost of:

- Session allocation, initiation of the mirror transaction, termination of the mirror transaction, and session de-allocation.

For example, if you were migrating from a local file access to MRO XM and requesting 6 function ships per transaction, the additional cost could be calculated as follows:

13.2(Initiate/Terminate)+6(requests)*9.0(Request Cost)+ 9.0(Syncpoint) = 76.0

Function shipping (MROLRM=NO)

Without long running mirrors each function ship read request incurs the cost of session allocation and mirror initialization and termination. However, the first change to a protected resource (for example, a READ UPDATE or a WRITE) causes the session and mirror to be held until a syncpoint.

MRO XM	MRO XCF (via CTC)	MRO XCF (via CF)	ISC LU6.2
21.4	35.0	59.9	115.0

Glossary

This glossary defines CICS terms used in this book and words used with other than their everyday meaning. In some cases, a definition may not be the only one applicable to a term, but gives the particular sense in which it is used in this book.

This glossary includes terms and definitions from the *IBM Vocabulary for Data Processing, Telecommunications, and Office Systems*, GC20-1699.

American National Standards Institute (ANSI) definitions are preceded by an asterisk (*).

The symbol "(ISO)" at the beginning of a definition indicates that it has been discussed and agreed on at meetings of the International Organization for Standardization, Technical Committee 97/Subcommittee 1, and has been approved by ANSI for inclusion in the *American National Dictionary for Information Processing*.

A

abend. Abnormal end of task.

ACB. See access method control block .

access key. An indicator associated with a reference to storage. It is matched to the **storage key** to permit access to the storage. In most cases, the access key is the PSW key in the current PSW.

access method. A technique for moving data between main storage and input/output devices.

access method control block (ACB). A control block that links an application program (for example, a CICS program) to an access method (for example VSAM or ACF/VTAM). An ACB is used when communicating with DL/I only when the underlying access method is VSAM.

ACF. See advanced communications function.

active session. In XRF, a session between a class 1 terminal and the active system. A session that connects the active CICS to an end user.

active system. In an XRF environment, the CICS system that currently supports the processing requests of the user.

active task. A CICS task that is eligible for dispatching by CICS. During emergency restart, a task that completed a UOW and started another, but that did not cause any records to be written to the system log during the second UOW. During recovery-control processing, a UOW completion but no physical end-of-task (that is, task DETACH) is found.

activity keypoint. A record of task and DCT entry status on the system log made on a periodic basis to facilitate the identification of transaction backout information during emergency restart. In the event of an uncontrolled shutdown and subsequent emergency restart, activity keypoints can shorten the process of backward scanning through the system log. Activity keypoints are written automatically by the system (**system activity keypoints**) or by the user (**user activity keypoints**).

address space. The complete range of addresses that is available to a program.

addressing mode (AMODE). In MVS, the mode, 24-bit or 31-bit, in which a program stores addresses. The AMODE linkage-editor control statement specifies the addressing mode of the load module produced.

Advanced Communications Function (ACF). A group of program products for users of MVS that can improve single-domain and, optionally, multidomain data communication capability.

Advanced Program-to-Program Communication (APPC). The SNA protocol boundary of the presentation services layer of the LU6.2 architecture. APPC is commonly used as a synonym for LU6.2.

after image. A record of the contents of a data element after it has been changed. After images are used for forward recovery.

agent. In a two-phase commit or MRO syncpointing sequence, a task that receives syncpoint requests from the initiator (the task that initiates the syncpoint activity).

AID. Automatic Initiate Descriptor.

AIEXIT. System initialization parameter used to specify the name of the autoinstall user program that you want CICS to use when autoinstalling VTAM terminals. The default is the name of the CICS-supplied autoinstall user program, DFHZATDX. See the *CICS System Definition Guide* for more information.

AILDELAY. System initialization parameter used to specify the delay period that elapses between the end of a session between CICS and a terminal and the deletion of the terminal entry. The default is zero,

meaning that the terminal entry is deleted as soon as the session is ended. See the *CICS System Definition Guide* for more information.

AIQMAX. System initialization parameter used to specify the maximum number of devices that can be queued concurrently for autoinstall. The default is 1000. See the *CICS System Definition Guide* for more information.

AIRDELAY. System initialization parameter used to specify the delay period that elapses after an emergency restart, before autoinstalled terminals that are not in session are deleted. The default is 700, meaning a delay of seven minutes. See the *CICS System Definition Guide* for more information.

AKPFREQ. System initialization parameter used to specify the frequency of activity keypoints. The default is 1000. See the *CICS System Definition Guide* for more information.

alternate system. In an XRF environment, a CICS system that stands by to take over the user workload when the active CICS system fails or a takeover is initiated.

AMODE. See addressing mode.

APAR. Authorized program analysis report.

APPC. See advanced program-to-program communication.

APPLID. System initialization parameter used to specify the VTAM application identifiers (applids) for this CICS region. See the *CICS System Definition Guide* for more information.

assembler language. A source language that includes symbolic machine-language statements in which there is a one-to-one correspondence with the instruction formats and the data formats of the computer. Before execution, a CICS assembler-language application program must be processed by the translator, assembler, and linkage editor.

asynchronous processing. A means of distributing the processing of an application between systems in an intercommunication environment. The processing in each system is independent of the session on which requests are sent and replies are received. No direct correlation can be made between requests and replies and no assumptions can be made about the timing of the replies.

automatic initiate descriptor (AID). A control block used internally by CICS for scheduling purposes. An example of AID use is scheduling a transaction, optionally associating it with a terminal and a temporary storage queue. Another use is scheduling MRO, LU6.1, and LU6.2 ALLOCATE requests.

automatic transaction initiation (ATI). The initiation of a CICS transaction by an internally-generated request, for example, the issue of an EXEC CICS START command or the reaching of a transient data trigger level. A transaction can be initiated immediately, at a specified time, after a specified time interval, or, if a terminal is required, as soon as that terminal is free.

auxiliary storage. Data storage other than main storage; for example, storage on magnetic tape or direct access devices.

auxiliary trace. An optional CICS function that causes trace entries to be recorded in the auxiliary trace data set, a sequential data set on disk or tape.

AUXTR. System initialization parameter used to indicate whether the auxiliary trace destination is to be activated at system initialization. This parameter controls whether any of the three types of CICS trace entry (system trace, user trace, and exception trace) are written to the auxiliary trace data set. See the *CICS System Definition Guide* for more information.

availability. The degree to which a system or resource is ready when needed to process data; the *percentage* of time a system, network, or component can be utilized, within a certain time frame. Generally, the percentage is derived by dividing **actual availability** time by **scheduled availability** time.

average throughput rate. The power of a system to process a representative work load. The power of the system is measured in units of data processing work, for example, jobs or transactions successfully completed per hour, minute, or second.

AVM. Availability manager.

AXM. The 'authorized cross-memory' server environment, a set of modules providing run-time services for CICS-related cross-memory servers which run in MVS authorized state (unlike CICS itself, which runs unauthorized once initialization has completed) such as the temporary storage data sharing server.

B

backout. The process of restoring to a previous state all or part of a system. The process of removing all the updates against **protected resources** such as files and DL/I databases performed by an application program that either has terminated abnormally or was inflight at the time of a CICS or MVS image failure. Backout can by done dynamically in the case of an application abend, or during restart in the case of CICS or MVS failure.

backup session. In XRF, the session built by VTAM to the alternate CICS system for XRF-capable terminals, used after a takeover to reestablish service to the terminals.

Basic Direct Access Method (BDAM). An access method used to retrieve or update particular blocks of a data set on a direct access device.

basic mapping support (BMS). A facility that moves data streams to and from a terminal. BMS is an interface between CICS and its application programs. It formats input and output display data in response to BMS commands in programs. To do this, it uses device information from CICS system tables and formatting information from maps you have prepared for your application programs.

BMS provides message routing, terminal paging, and device independent services. Most of the BMS programs are resident in the CICS nucleus.

BMS exists in three pregenerated versions: minimum, standard, and full function. Each version provides a different level of function, and therefore

Basic sequential access method (BSAM). An access method for storing or retrieving data blocks in a continuous sequence.

BTS. CICS business transaction services

Basic telecommunications access method (BTAM). An access method that enables read/write communication with remote devices.

BDAM. Basic direct access method.

Binary synchronous communication (BSC). Data transmission in which synchronization of characters is controlled by timing signals generated at the sending and receiving stations.

BookManager. A family of IBM products that enable users to create and display online books.

boundary network node (BNN). In SNA, a subarea node that provides protocol support for adjacent peripheral nodes, for example, transforming network addresses to local addresses and *vice versa*, and providing session-level support for these peripheral nodes. In XRF, the point at which terminal sessions are switched from the failing active system to the new active system. The communication controller at the BNN must be able to operate in an XRF configuration.

BSAM. See Basic sequential access method.

BSC. Binary synchronous communication.

C

capacity planning. An analysis of processor loading and processor capacity, extending into real storage, other resources (channels, DASD, lines), and timings and response when necessary.

CAVM. See CICS availability manager.

CDSA. See CICS dynamic storage area.

CDSASZE. System initialization parameter used to specify the amount of storage to be allocated by CICS for the CICS dynamic storage area (CDSA) below the 16MB line. The default size is 1536KB. See the *CICS System Definition Guide* for more information.

CEBR. A CICS-supplied transaction used to browse temporary storage queues. See the *CICS Supplied Transactions* manual for more information.

CEBT. A CICS-supplied transaction that the operator can issue from the MVS console to control an alternate CICS system. See the *CICS Supplied Transactions* manual for more information.

CEDA. The main CICS-supplied transaction used to define resources online. Using CEDA, you can update both the CICS system definition data set (CSD) and the running CICS system. See the *CICS Resource Definition Guide* for more information.

CEMT. A CICS-supplied transaction used to invoke all the master terminal functions. These functions include inquiring and changing the value of parameters used by CICS, altering the status of system resources, terminating tasks, and shutting down CICS. See the *CICS Supplied Transactions* manual for more information.

CEST. A CICS-supplied transaction used to invoke a subset of the master terminal (CEMT) functions. CEST allows you to inquire about and alter some of the values of lines, netnames, tasks, and terminals. See the *CICS Supplied Transactions* manual for more information.

CETR. A CICS-supplied transaction used to control CICS tracing activity. See the *CICS Supplied Transactions* manual for more information.

checkpoint. In IMS, point at which an application program commits that the changes it has made to a database are consistent and complete, and releases database segments for use by other programs. You can request checkpoints at appropriate points in a program to provide places from which you can restart that program if it, or the system, fails.

For an IMS system, a point in time from which the system can start again if a failure makes recovery necessary. The checkpoint is performed by IMS itself.

CI. See Control interval.

CICS availability manager (CAVM). In XRF, the mechanism that provides integrity for a CICS system with XRF. The CAVM uses the control file and the message file to handle communication between the active and alternate systems.

CICS business transaction services. CICS domains that support an application programming interface (API) and services that simplify the development of business transactions.

CICS dynamic storage area (CDSA). A storage area allocated below the 16MB line, intended primarily for the small amount of CICS code and control blocks that remain below the line in CICS/ESA 3.3. The size of the CDSA is controlled by the CDSASZE system initialization parameter.

CICS monitoring facility. The CICS monitoring facility (part of the system monitoring component) gives a comprehensive set of operational data for CICS, using one data recording program and, optionally, one or more data sets. See also **performance class data**, **exception class data**, and **SYSEVENT data**.

CICS monitoring facility data set. CICS monitoring facility data sets are used to record information that is output by the CICS monitoring facility program. The MCT defines which journal data sets are used by each class of monitoring. These data sets appear in the JCT with the FORMAT=SMF parameter. The format of the records is the system management facility (SMF), type 110 format.

CICS PD/MVS. CICS Problem Determination/MVS (program number 5695-035) is a set of online tools to help system programmers analyze and manage system dumps. It automates dump analysis and formats the results into interactive online panels that can be used for further diagnosis and resolution of problems.

CICS private area. Element of CICS storage that has both static and dynamic storage requirements. The static areas are set at initialization time and do not vary over the execution of that address space. The dynamic areas increase or decrease their allocations as the needs of the address space vary.

CICS program library. The CICS program library contains all user-written programs and CICS programs to be loaded and executed as part of the online system. This group includes the control system itself and certain user-defined system control tables essential to CICS operation. The library contains program text and, where applicable, a relocation dictionary for a program. The contents of this library are loaded asynchronously into CICS dynamic storage for online execution.

CICS region userid. The userid assigned to a CICS region at CICS initialization. It is specified **either** in the RACF started procedures table when CICS is started as a started task, **or** on the USER parameter of the JOB statement when CICS is started as a job.

CICS system definition data set (CSD). A VSAM KSDS cluster with alternate paths. The CSD data set contains a resource definition record for every record defined to CICS using resource definition online.

CICS-attachment facility. Provides a multithread connection to DB2 to allow applications running under CICS to execute DB2 commands.

CICS-key. Storage in the key in which CICS is given control (key 8) when CICS storage protection is used. It is for CICS code and control blocks and can be accessed and modified by CICS. Application programs in user-key cannot modify CICS-key storage, but they can read it. The storage is obtained in MVS exclusive-key storage. Compare with **user-key**..

CICSPARS/MVS. The Customer Information Control System Performance Analysis Reporting System (CICSPARS/MVS) (program number 5665-355) provides a method of reporting performance and accounting information produced by the CICS monitoring facility.

class 1 terminal. In XRF, a remote SNA VTAM terminal connected through a boundary network node IBM 3745/3725/3720 Communication Controller with an NCP that supports XRF. Such a terminal has a backup session to the alternate CICS system.

class 2 terminal. In XRF, a terminal belonging to a class mainly comprised of VTAM terminals that are not eligible for class 1. For these terminals, the alternate system tracks the session, and attempts reestablishment after takeover.

class 3 terminal. In XRF, a terminal belonging to a class mainly comprised of TCAM(DCB) terminals. These terminals lose their sessions at takeover.

CLT. System initialization parameter used to specify the suffix for the command list table, if this system initialization table is used by an alternate XRF system. See the *CICS System Definition Guide* for more information.

cluster. A data set defined to VSAM. A cluster can be a key-sequenced data set, an entry-sequenced data set, or a relative record data set.

CMAC. A CICS-supplied transaction used to display individual message information as it is provided in the CICS *CICS Messages and Codes* manual. See the *CICS Supplied Transactions* manual for more information.

COBOL. Common business-oriented language. An English-like programming language designed for business data processing applications.

command list table (CLT). In XRF, a CICS table that contains a list of MVS commands and messages to be issued during takeover. The CLT is defined to the alternate CICS system and used during takeover.

command security. A form of security checking that can be specified for a subset of the CICS application programming interface (API) commands. Command security operates in addition to any transaction security or resource security specified for a transaction. For

example if a terminal invokes a transaction that the user is authorized to use, and the transaction issues a command that the user is not authorized to use, the command fails with the NOTAUTH condition.

COMMAREA. See Communication area.

common system area (CSA). A major CICS storage control block that contains areas and data required for the operation of CICS. It can be extended to include a user-defined common work area (CWA) that can be referred to by application programs. This area is duplicated above the 16MB line as the **extended common system area (ECSA)**.

common work area (CWA). The common work area (CWA) is an area within the CSA that can be used by application programs for user data that needs to be accessed by any task in the system. This area is acquired during system initialization and its size is determined by the system programmer at system generation. It is initially set to binary zeros. Its contents can be accessed and altered by any task during CICS operation. Contrast with **transaction work area (TWA)**.

communication area (COMMAREA). An area that is used to pass data between tasks that communicate with a given terminal. The area can also be used to pass data between programs within a task.

communication management configuration (CMC). A configuration in which the VTAM subsystem that owns the terminals is in a different MVS image from the active or the alternate CICS system.

constraint. This is sometimes referred to as "transaction throughput degradation" or "bottleneck"—a place in the system where contention for a resource is affecting performance.

control block. In CICS, a storage area used to hold dynamic data during the execution of control programs and application programs. Synonym for **control area**. Contrast with **control table**.

control data set. A data set that ensures XRF system integrity by allowing only one active CICS system to access a particular set of resources. It is used by the active and the alternate CICS systems to monitor each other's well-being.

control interval (CI). A fixed-length area of auxiliary-storage space in which VSAM stores records and distributes free space. The unit of information transmitted to or from auxiliary storage by VSAM, independent of physical record size.

control subpool. A CICS area that holds the dispatch control area (DCA), interval control elements (ICEs), automatic initiate descriptors (AIDs), queue element areas (QEAs), and other control information. Generally, the control subpool occupies only one page.

control table. In CICS, a storage area used to define or describe the configuration or operation of the system. Contrast with **control block**.

control terminal. In CICS, the terminal at which a designated control operator is signed-on.

conversational. Pertaining to a program or a system that carries on a dialog with a terminal user, alternately accepting input and responding to the input quickly enough for the user to maintain a train of thought.

CSA. See Common system area.

CSAC. Transient data destination used by the abnormal condition program (DFHACP).

CSD. See CICS system definition data set.

CSMT. Transient data destination used by the terminal abnormal condition program (DFHTACP), the node abnormal condition program (DFHZNAC), and the abnormal condition program (DFHACP) for writing terminal error and abend messages.

CSTE. Transient data destination used by the terminal abnormal condition program (DFHTACP).

CWA. See Common work area.

CWAKEY. System initialization parameter used to specify the storage key for the CWA if CICS is running with the storage protection facility. The default storage key is user-key.

D

DASD. Direct access storage device.

data availability. An IMS enhancement available with DBCTL. It allows PSB scheduling to complete successfully even if some of the full-function databases it requires are not available.

Data control block (DCB). An MVS control block used by access method routines in storing and retrieving data.

data element. The smallest unit of data that can be referred to. Synonymous with **field**.

data entry database (DEDB). An IMS hierarchic database designed to provide efficient storage and fast online gathering, retrieval, and update of data using VSAM ESDS. From CICS, a DEDB is accessible only through DBCTL, not through local DL/I.

Data Language/I (DL/I). A high-level interface between applications and IMS. It is invoked from PL/I, COBOL, or assembler language by means of ordinary subroutine calls. DL/I enables you to define data structures, to relate structures to the application, and to

load and reorganize these structures. It enables applications programs to retrieve, replace, delete, and add segments to databases.

data management block (DMB). An IMS control block that resides in main storage and describes and controls a physical database. It is constructed from information obtained from the application control block (ACB) library or the database description (DBD) library.

data set name sharing. An MVS option that allows one set of control blocks to be used for the base and the path in a VSAM alternate index.

data sharing (IMS). The concurrent access of DL/I databases by two or more IMS/VS subsystems. The subsystems can be in one processor or in separate processors. In IMS data sharing, CICS can be an IMS subsystem. There are two levels of data sharing: **block-level data sharing** and **database-level data sharing**.

data stream. All information (data and control information) transmitted through a data channel in a single read or write operation.

data-owning region (DOR). A CICS address space whose primary purpose is to manage files and databases. See **application-owning region (AOR)**, and **terminal-owning region (TOR)**.

DATABASE 2 (DB2). A relational database management system in which data is presented to the user in the form of tables.

database-level sharing. A kind of IMS data sharing that enables application programs in one IMS system to read data while a program in another IMS system reads it or updates it.

DCT. System initialization parameter used to specify the destination control table suffix. See Destination control table. For more information, see the *CICS System Definition Guide*.

deadlock. Unresolved contention for the use of a resource. An error condition in which processing cannot continue because each of two elements of the process is waiting for an action by, or a response from, the other.

deferred work element (DWE). A work element created and placed on a chain (the DWE chain) to save information about an event that must be completed before task termination but is not completed at the present time. DWEs are also used to save information about work to be backed out in case of an abend.

destination control table (DCT). A table containing an entry for each extrapartition, intrapartition, and indirect transient data destination used in the system, or in connected CICS systems.

device independence. The capability to write application programs so that they do not depend on the physical characteristics of devices. BMS provides a measure of device independence.

DFH. Three-character prefix of all CICS modules.

DFHCSDUP. CICS system definition data set (CSD) utility program. It provides offline services for the CSD. It can be invoked as a batch program or from a user-written program running either in batch mode or under TSO.

dispatching. The act of scheduling a task for execution, performed by CICS task control.

dispatching priority. A number assigned to tasks, used to determine the order in which they are to use the processor in the CICS multitasking environment.

distributed program link (DPL). Type of CICS intercommunication which, in CICS/ESA 3.3, enables CICS to ship LINK requests between host CICS regions. In CICS OS/2, DPL enables CICS OS/2™ to ship LINK requests up to a host CICS region, or to another CICS OS/2 system.

distributed transaction processing (DTP). Type of intercommunication in CICS, in which the processing is distributed between transactions that communicate synchronously with one another over intersystem or interregion links.

DMB. See Data management block (DL/I).

DPL. Distributed program link.

DSALIM. Dynamic storage area. System initialization parameter.

DTB. Dynamic transaction backout.

DTP. Distributed transaction processing.

DUMP. System initialization parameter used to specify whether CICS is to take SDUMPs. The default is YES. See the *CICS System Definition Guide* for more information.

dump control. The CICS element that provides storage dumps for help during testing.

dynamic transaction backout (DTB). The process of canceling changes made by a transaction to stored data following the failure of that transaction for whatever reason. Dynamic transaction backout is required with resource definition online.

E

EDSALIM. Extended dynamic storage area. System initialization parameter.

emergency restart. The CICS backout facility for an automatic restart following a system failure. It restores the recoverable resources of all interrupted transactions to the condition they were in when they started.

end-of-day statistics. CICS statistics collected since the last event that involved a reset, such as a shutdown.

enqueued. The state of a task scheduled to update a physical segment of a database when another task is currently accessing that segment.

Event control block (ECB). An MVS control block that represents the status of an event. CICS task control uses ECBs.

event monitoring point (EMP). Point in the CICS code at which CICS monitoring data is collected. You cannot relocate these system-defined points.

exception class data. CICS monitoring information on exception conditions raised by a transaction, such as queuing for VSAM strings or waiting for temporary storage. This data highlights possible problems in system operations.

exception trace entry. An entry made to the internal trace table and any other active trace destinations when CICS detects an exception condition. It gives information about what was happening at the time the failure occurred and what was being used.

exclusive-key storage. In MVS key-controlled storage protection, storage with storage keys other than open-key.

EXEC. Key word used in CICS command language. All CICS commands begin with the keywords `EXEC CICS`.

Execution Diagnostic Facility (EDF). A facility used for testing application programs interactively online, without making any modifications to the source program or to the program preparation procedure. The facility intercepts execution of the program at various points and displays information about the program at these points. Also displayed are any screens sent by the user program, so that the programmer can converse with the application program during testing just as a user would do on the production system.

extended CICS dynamic storage area (ECDSA). Storage area allocated above the 16MB line for CICS code and control blocks which are eligible to reside above the 16MB line, but are not eligible for the ERDSA (that is, they are not reentrant.) See the *CICS System Definition Guide* for more information.

extended common system area (ECSA). A major element of MVS/ESA virtual storage above the 16MB line. This area contains pageable system data areas that are addressable by all active virtual storage address spaces. It duplicates the **common system area (CSA)** which exists below the 16MB line.

extended error queue element (EEQE). Contains data describing I/O errors on a database recorded by CICS.

extended link pack area (ELPA). A major element of MVS/ESA virtual storage above the 16MB line. It duplicates the **link pack area (LPA)**. See also **extended addressing**.

extended private area. An element of MVS/ESA virtual storage above the 16MB line. This area duplicates the **private area** except for the 16KB system region area.

extended read-only dynamic storage area (ERDSA). An area of storage allocated above the 16MB line and used for eligible, reentrant CICS and user application programs, which must be link-edited with the RENT and AMODE(31) attributes. The storage is obtained in key 0, non-fetch-protected storage.

Extended Recovery Facility (XRF). A facility that increases the availability of CICS transaction processing, as seen by the end users. Availability is improved by having a second CICS system (the **alternate system**) ready to continue processing the workload, if and when particular failures that disrupt user services occur on the first system (the **active system**).

Extended Recovery Facility (XRF) complex. All the required hardware and software components (MVS images and licensed programs) that provide the XRF function for a CICS system.

extended system queue area (ESQA). A major element of MVS/ESA virtual storage above the 16MB line. This storage area contains tables and queues relating to the entire system. It duplicates above the 16MB line the **system queue area (SQA)**.

extended user dynamic storage area (EUDSA). Storage area allocated above the 16MB line and reserved exclusively for those user application programs that execute in user-key, that are eligible to reside above the 16MB line, but are not eligible for the ERDSA (that is, they are not reentrant.)

external response time. Elapsed time from pressing the ENTER key or another AID key until the action requested by the terminal user is completed, and the next entry can be started.

external security manager (ESM). A program, such as RACF, that performs security checking for CICS users and resources.

external throughput rate (ETR). The amount of useful work completed in a unit of time (for example, the number of transactions completed per elapsed second).

extrapartition transient data. A CICS facility for temporarily saving data in the form of queues, called destinations. Each extrapartition TD destination requires a resource definition that links it to a QSAM data set outside the CICS region. Each extrapartition TD destination uses a different QSAM data set. Extrapartition destinations are used for data that is either coming from a source outside the region, or being directed from a source within the region to a destination outside the region. Extrapartition data written by CICS is usually intended for subsequent input to non-CICS batch programs. Examples of data that might be written to extrapartition destinations include logging records, statistics, and transaction error messages. Contrast with **intrapartition transient data**.

F

FCT. System initialization parameter used to specify the suffix of the file control table to be used. This parameter is effective only on a CICS cold or initial start. See the *CICS System Definition Guide* for more information.

file control table (FCT). Table containing the characteristics of the files accessed by file control.

file request thread element (FRTE). An element used by CICS file control to link related requests together as a file thread; to record the existence of READ SET storage to be released at syncpoint and the existence of any other outstanding work that must be completed at syncpoint; to register a task as a user of a file to prevent it being closed while still in use.

file-owning region (FOR). A CICS address space whose primary purpose is to manage files and databases. Deprecated term for **data-owning region (DOR)**. See also **application-owning region (AOR)**, and **terminal-owning region (TOR)**.

first failure data capture (FFDC). Data relevant to a CICS exception condition that is recorded as soon as possible after the condition has been detected.

fixed-block-architecture (FBA) device. A disk storage device that stores data in blocks of fixed size. These blocks are addressed by block number relative to the beginning of the particular file.

format. The arrangement or layout of data on a data medium, usually a display screen with CICS.

format independence. The ability to send data to a device without having to be concerned with the format in which the data is displayed. The same data may appear in different formats on different devices.

fragmentation. The breaking up of free storage into small areas (by intervening used storage areas). This leads to the effective storage available for use being reduced.

FREEMAIN. EXEC CICS command used to release main storage. For programming information, see the *CICS Application Programming Reference* manual.

function shipping. The process, transparent to the application program, by which CICS accesses resources when those resources are actually held on another CICS system.

G

GTF. Generalized trace facility—a data-collection routine in MVS. GTF traces the following system events: seek addresses on start I/O records, SRM activity, page faults, I/O activity, and supervisor services. Execution options specify the system events to be traced.

GCD. Global catalog data set. Global catalog is a VSAM key-sequenced data set (KSDS). Essential for recovery purposes.

H

high performance option (HPO). A option provided with MVS to improve performance by reducing the transaction pathlength; that is, the number of instructions needed to service each request.

high private area. Part of the CICS address space, consisting of the local system queue area (LSQA), the scheduler work area (SWA), and subpools 229 and 230. The area at the high end of the CICS address space is not specifically used by CICS, but contains information and control blocks that are needed by the operating system to support the region and its requirements.

Hiperspace. A high-performance storage area in the processor or multiprocessor.

host computer. The primary or controlling computer in a data communication system.

host processor. The primary or controlling computer in a multiple computer installation.

HPO. System initialization parameter used to indicate whether you want to use the VTAM authorized path feature of the high performance option. The default is NO. You can code this parameter only in the system initialization table. See the *CICS System Definition Guide* for more information.

I

I/O. Input/output (primarily from and to terminals)

ICP. System initialization parameter used to specify that you want to cold start the CICS interval control program. See the *CICS System Definition Guide* for more information.

ICV. System initialization parameter used to specify the region exit interval time in milliseconds. The region exit interval is the maximum interval of time for which CICS releases control to the operating system in the event that there are no transactions ready to resume processing. The default interval is 1000 milliseconds. See the *CICS System Definition Guide* for more information.

ICVR. System initialization parameter used to specify the runaway task time interval in milliseconds as a decimal number. CICS purges a task if it has not given up control after this length of time (that is, if the task appears to be looping). The default value is 5000 milliseconds. See the *CICS System Definition Guide* for more information.

ICVTSD. System initialization parameter used to specify the terminal scan delay. The terminal scan delay facility determines how quickly CICS deals with some terminal I/O requests made by applications. The default value is 500 milliseconds. See the *CICS System Definition Guide* for more information.

in-flight task. A task that is in progress when a CICS system failure or immediate shutdown occurs. During emergency restart, a task that caused records to be written to the system log, but for which no syncpoint record has been found for the current UOW. This task was interrupted before the UOW completed.

in-flight transaction. Any transaction that was still in process when system termination occurred.

Information Management System (IMS). A database manager used by CICS to allow access to data in DL/I databases. IMS provides for the arrangement of data in an hierarchical structure and a common access approach in application programs that manipulate IMS databases.

initialization. Actions performed by the CICS system to construct the environment in the CICS region to enable CICS applications to be run. The stage of the XRF process when the active or the alternate CICS system is started, signs on to the control data set, and begins to issue its surveillance signal.

initialization phase. The process of bringing up the active CICS system and the alternate CICS system in an XRF complex. The two actions are performed independently.

installation. A particular computing system, in terms of the work it does and the people who manage it, operate it, apply it to problems, service it, and use the work it produces. The task of making a program ready to do useful work. This task includes generating a program, initializing it, and applying any changes to it.

interactive. Pertaining to an application in which each entry entails a response from a system or program, as in an inquiry system or an airline reservation system. An interactive system may also be conversational, implying a continuous dialogue between the user and the system.

intercommunication facilities. A term covering intersystem communication (ISC) and multiregion operation (MRO).

intermediate routing node (IRN). A subarea node, which may receive and route sessions that neither originate in nor are destined for network addressable units in that subarea node. Terminals attached to an IRN cannot have XRF backup sessions.

internal response time. Elapsed time from the message to start a transaction being received by CICS until the time that the transaction ends.

internal throughput rate (ITR). The number of completed transactions per processor-busy second. (Processor busy seconds can be calculated by multiplying elapsed seconds by the processor utilization percentage).

internal trace. CICS trace facility that is always present in virtual storage. When CICS detects an exception condition, an entry always goes to the internal trace table, even if you have turned tracing off. The internal trace table is a wraparound table whose size can be set by the TRTABSZ system initialization parameter and can be changed by the CICS SET TRACE DEST command. See the *CICS Problem Determination Guide* for more information.

interregion communication (IRC). The method by which CICS provides communication between a CICS region and another region in the same processor. Used for **multiregion operation (MRO)** . Compare with **intersystem communication**.

intersystem communication (ISC). Communication between separate systems by means of SNA networking facilities or by means of the application-to-application facilities of VTAM. ISC links CICS systems and other systems, and may be used for user application to user application communication, or for transparently executing CICS functions on a remote CICS system. Compare with **multiregion operation** and **interregion communication**.

interval control element (ICE). An element created for each time-dependent request received by the interval control program. These ICEs are logically chained to the CSA in expiration time-of-day sequence.

Expiration of a time-ordered request is detected by the expired request logic of the interval control program running as a CICS system task whenever the task dispatcher gains control. The type of service represented by the expired ICE is initiated, providing all resources required for the service are available, and the ICE is removed from the chain. If the resources are not available, the ICE remains on the chain and another

attempt to initiate the requested service is made the next time the task dispatcher gains control.

interval control program (ICP). The CICS program that provides time-dependent facilities. Together with task control, interval control (sometimes called time management) provides various optional task functions (system stall detection, runaway task control, task synchronization, etc.) based on specified intervals of time, or the time of day.

interval statistics. CICS statistics gathered during a specified interval. See also **end-of-day statistics**, **requested statistics**, **requested reset statistics**, and **unsolicited statistics**.

intrapartition transient data (TD). A CICS facility for temporarily saving data in the form of queues, called destinations. All intrapartition TD destinations are held as queues in the same VSAM data set, which is managed by CICS. Data is written to the queue by a user task. The queue can be used subsequently as input data by other tasks within the CICS region. All access is sequential, governed by read and write pointers. Once a record has been read it cannot be read subsequently by another task. An intrapartition destination requires a resource definition containing information that locates the queue in the intrapartition data set. Applications that might use intrapartition queues include message switching, data collection, and queuing of orders.

ISC. System initialization parameter used to include the CICS programs required for interregion or intersystem communication. See the *CICS System Definition Guide* for more information.

J

job control language (JCL). Control language used to describe a job and its requirements to an operating system.

K

key 0. Storage used by CICS for the extended read-only dynamic storage area (ERDSA). The allocation of key 0 storage for the ERDSA is optional.

key-controlled storage protection. An MVS facility for protecting access to storage. Access to key-controlled storage is permitted only when the **storage key** matches the **access key** associated with the request.

keypoint. The periodic recording of system information and control blocks on the system log—also the data so recorded. See also **activity keypoint**, and **warm keypoint**.

L

Last-in-first-out (LIFO). A queuing technique in which the next item to be retrieved is the last item placed in the queue.

LIFO storage. Storage used by reentrant CICS management modules to save registers.

link pack area (LPA). A major element of MVS/ESA virtual storage below the 16MB line. The storage areas that make up the LPA contain all the common reentrant modules shared by the system, and exists to provide economy of real storage by sharing one copy of the modules, protection because LPA code cannot be overwritten even by key 0 programs, and reduced pathlength because the modules can be branched to. The LPA is duplicated above the 16MB line as the **extended link pack area (ELPA)**.

LISTCAT. A VSAM tool that provides information that interprets the actual situation of VSAM data sets.

local. In data communication, pertaining to devices that are accessed directly without use of a telecommunication line. Contrast with **remote**. Synonym for channel-attached.

local DL/I. DL/I residing in the CICS address space.

local resource. In CICS intercommunication, a resource that is owned by the local system.

local shared resources (LSR). Files that share a common pool of buffers and a common pool of strings; that is, control blocks supporting the I/O operations. Contrast with **nonshared resources**.

local system. The system in a multisystem environment on which the application program is executing. The local application may process data from databases located on both the same (local) system and another (remote) system. Contrast with **remote system**.

local system queue area (LSQA). An element of the CICS address space. It generally contains the control blocks for storage and contents supervision. See also **high private area**.

local work area. Area provided for the use of a single task-related user exit program. It is associated with a single task and lasts for the duration of the task only.

log. A recording of changes made to a file. This recording can be used for subsequent recovery of the file. See also **dynamic log**, **journal**, and **system log**.

logging. The recording (by CICS) of recovery information onto the system log, for use during emergency restart. A specific journaling function that records changes made to the system activity environment and database environment. These records

are required for recovery and backout support by CICS (and the user) following an abnormal termination.

logical partition (LP). A partition, in a CEC, capable of running its own MVS image. It comprises a set of hardware resources (processors, storage, channels, and so on), sufficient to allow a system control program such as MVS to be executed.

logical unit (LU). In SNA, a port through which a user gains access to the services of a network.

look-aside query. Query performed in one partition by an operator working in another partition. Using partitions, a partially completed operation need not be transmitted to the host processor before releasing the screen for an inquiry.

LPA. System initialization parameter used to indicate whether any CICS management modules can be used from the link pack area. The default is NO. See the *CICS System Definition Guide* for more information.

LUTYPE6.1 (LU6.1). Type of logical unit used for processor-to-processor sessions. LUTYPE6.1 is a development of LUTYPE6. CICS—IMS intercommunication uses LUTYPE6.1 sessions.

LUTYPE6.2 (LU6.2). Type of logical unit used for CICS intersystem (ISC) sessions. LUTYPE6.2 is a development of LUTYPE6.1. The LUTYPE6.2 architecture supports both CICS host to system-level products and CICS host to device-level products. CICS ISC uses LUTYPE6.2 sessions. APPC is the the protocol boundary of the LU6.2 architecture.

M

main storage. (ISO) Program-addressable storage from which instructions and data can be loaded directly into registers for subsequent execution or processing. See also **real storage, storage, virtual storage**.

map. A format established for a page or a portion of a page, or a set of screen format descriptions. A concept of CICS BMS that maps the relationship between the names of program variables and the position in which their values will appear on a display device. The map also contains other formatting information such as field attributes. A map describes constant fields, and their position on the display; the format of input and output fields; the attributes of constant and variable fields and the symbolic names of variable fields.

message control program (MCP). In ACF/TCAM, a specific implementation of an access method, including I/O routines, buffering routines, activation and deactivation routines, service facilities, and SNA support.

message data set. In XRF, a data set used by the active CICS system to transmit messages to the alternate CICS system.

message performance option. The improvement of ISC performance by eliminating syncpoint coordination between the connected systems.

message switching. In a data network, the process of routing messages by receiving, storing, and forwarding complete messages.

mirror task. A task required to service any incoming request that specifies a CICS mirror transaction (CSMI, CSM1, CSM2, CSM3, CSM5, CPMI, CVMI).

mirror transaction. Recreates the request that is function shipped from one system to another, issues the request on the second system, and passes the acquired data back to the first system.

MN. System initialization parameter used to indicate whether monitoring is to be switched on or off at initialization. The default is OFF. See the *CICS System Definition Guide* for more information.

MNEVE. System initialization parameter used to indicate whether SYSEVENT monitoring is to be made active during CICS initialization. The default is OFF. See the *CICS System Definition Guide* for more information.

MNEXC. System initialization parameter used to indicate whether the monitoring exception class is to be made active during CICS initialization. The default is OFF. See the *CICS System Definition Guide* for more information.

MNPER. System initialization parameter used to indicate whether the monitoring performance class is to be made active during CICS initialization. The default is OFF. See the *CICS System Definition Guide* for more information.

modegroup. A VTAM LOGMODE entry, which can specify (among other things) the class of service required for a group of APPC sessions.

modename. The name of a modeset and of the corresponding modegroup.

modeset. In CICS, a group of APPC sessions. A modeset is linked by its modename to a modegroup (VTAM LOGMODE entry) that defines the class of service for the modeset.

modified link pack area (MLPA). An element of MVS/ESA virtual storage. This area provides a temporary extension to the PLPA existing only for the life of the current IPL. You can use this area to add or replace altered LPA-eligible modules without having to recreate the LPA. See also **link pack area (LPA)** and **pageable link pack area (PLPA)**.

monitoring. The regular assessment of an ongoing production system against defined thresholds to check that the system is operating correctly. Running a hardware or software tool to measure the performance characteristics of a system. Note that CICS distinguishes between monitoring and statistics, but IMS does not. See also **statistics**.

monitoring control table (MCT). A table describing the way the user data fields in the accounting and performance class monitoring records are to be manipulated at each user event monitoring point (EMP). It also identifies the CICS user journals in which the data for each monitoring class is to be recorded. The MCT contains the definition of user event monitor points (EMPs), and specifies the journal data sets used to record the data.

monitoring record. Any of three types of task-related activity record (performance, event, and exception) built by the CICS monitoring domain. Monitoring records are available to the user for accounting, tuning, and capacity planning purposes.

MROBTCH. System initialization parameter used to specify the number of events that must occur before CICS is posted for dispatch due to the batching mechanism. The default is one. See the *CICS System Definition Guide* for more information.

MROLRM. System initialization parameter used to specify whether you want to establish an MRO long-running mirror task. The default is NO. See the *CICS System Definition Guide* for more information.

multiregion operation (MRO). Communication between CICS systems in the same processor without the use of SNA network facilities. This allows several CICS systems in different regions to communicate with each other, and to share resources such as files, terminals, temporary storage, and so on. Contrast with **intersystem communication**.

multitasking. Concurrent execution of application programs within a CICS region.

multithreading. Use, by several transactions, of a single copy of an application program.

MVS image. A single copy of the MVS operating system. This can be a physical processing system (such as an IBM 3090) that is partitioned into one or more processors, where each partition is capable of running under the control of a single MVS operating system. Alternatively, if you are running MVS with the processor resource/systems manager (PR/SM), an MVS image can consist of multiple logical partitions, with each logical partition (LP) operating a copy of MVS. Also referred to as a single- or multi-MVS environment, according to the number of MVS systems.

MVS/DFP. MVS/Data Facility Product, a major element of MVS, including data access methods and data administration utilities.

MVS/ESA extended nucleus. A major element of MVS/ESA virtual storage. This area duplicates above the 16MB line the **MVS/ESA nucleus**.

MVS/ESA nucleus. A major element of MVS/ESA virtual storage. This static storage area contains control programs and key control blocks. The area includes the nucleus load module and is of variable size, depending on the installation's configuration. The nucleus is duplicated above the 16MB line as the **MVS/ESA extended nucleus**.

MXT. System initialization parameter used to specify the maximum number of tasks that CICS allows to exist at any time. The default is 32. See the *CICS System Definition Guide* for more information.

N

NEB. See Node error block.

NEP. See Node error program.

NETNAME (netname). In CICS, the name by which a CICS terminal or a CICS system is known to ACF/VTAM.

NetView. A network management product that can provide rapid notification of events and automated operations.

NetView Performance Monitor (NPM). A program product that collects and reports on data in the host and NCP.

network. An interconnected group of nodes. The assembly of equipment through which connections are made between data stations.

network configuration. In SNA, the group of links, nodes, machine features, devices, and programs that make up a data processing system, a network, or a communication system.

network control program (ACF/NCP). A program that controls the operation of a communication controller (3745, 3725, 3720, 3705) in which it resides. NCP builds the backup sessions to the alternate CICS system for XRF-capable terminals. NCP is generated by the user from a library of IBM-supplied modules.

Network Logic Data Manager (NLDM). A program that collects and interprets records of errors detected in a network and suggests possible solutions. NLDM consists of commands and data services processors that comprise the NetView software monitor component.

Network Performance Analysis and Reporting System (NETPARS). A program offering that analyzes network log data from the NetView Performance Monitor (NPM).

Network Problem Determination Application (NPDA). A program that collects and interprets records of errors detected in a network and suggests possible solutions. NPDA consists of commands and data services processors that comprise the NetView hardware monitor component.

NEWSIT. System initialization parameter used to cause CICS to load the specified system initialization table (SIT) and enforce the use of all SIT parameters, modified by any system initialization parameters provided through PARM, SYSIN, or the system console. You can code NEWSIT on PARM, SYSIN, or CONSOLE only. See the *CICS System Definition Guide* for more information.

node abnormal condition program (NACP). A CICS program used by terminal control to analyze terminal abnormal conditions that are logical unit or node errors detected by VTAM.

node error block (NEB). A set of recording areas of the node error table used to count node errors relating to a single logical unit.

node error program (NEP). A user-replaceable program used to allow user-dependent processing whenever a communication error is reported to CICS

nonconversational. A mode of CICS operation in which resources are allocated, used, and released immediately on completion of the task.

O

operating system. Software that controls the execution of programs; an operating system may provide services such as resource allocation, scheduling, input/output control, and data management.

P

page. In MVS/ESA, a fixed-length block that has a virtual address and that is transferred as a unit between real storage and auxiliary storage. The information displayed at the same time on a display device.

pageable link pack area (PLPA). An element of MVS/ESA virtual storage. This area contains supervisor call routines, access methods, and other read-only system programs along with read-only reenterable user programs selected by an installation to be shared among users of the system. Optional functions or devices selected by an installation during system generation add additional modules to the PLPA. See also **link pack area (LPA)** and **modified link pack area (MLPA)**.

paging. In MVS, the process of transferring pages between real storage and the external page storage known as the page data set.

performance. Together with ease-of-use (a measure of how easy it is to use a data processing system), a major factor on which the total productivity of a system depends. Performance is largely determined by a combination of three other factors: availability, response time, and throughput, and reflects the overall quality of service and operations of a given product or system.

performance analysis. The use of one or more performance tools to investigate the reasons for performance deterioration.

performance class data. Detailed transaction-level monitoring data, such as the processor and elapsed time for a transaction, or the time spent waiting for an I/O.

performance data section. Part of the CICS data section in a CICS monitoring record. It consists of a string of field connectors followed by one or more performance data records.

performance evaluation. The determination of how well a specific system is meeting or may be expected to meet specific processing requirements at specific interfaces. Performance evaluation, by determining such factors as throughput rate, turnaround time, and constrained resources, can provide important inputs and data for the performance improvement process.

performance improvement. The increase of the average throughput rate and operational capability, or the reduction of turnaround time.

Performance Reporter. Performance Reporter for MVS reporting system.

persistent verification (PV). The retention of a sign-on to a remote system across multiple conversations until it is no longer needed. The PVDELAY system initialization parameter defines how long entries can remain in the signed-on list in the remote system. See **PVDELAY**.

PL/I. A programming language designed for use in a wide range of commercial and scientific applications.

planned takeover. In XRF, a planned shutdown of the active CICS system, and takeover by the alternate system, for maintenance or operational reasons.

polling. The process whereby stations are invited, one at a time, to transmit. The polling process usually involves the sequential interrogation of several data stations.

post-takeover. The XRF phase, immediately following takeover, when the new active CICS system does not have an alternate system.

pregenerated system. A CICS system distributed in a form that has already undergone the system generation process.

priority. A rank assigned to a task that determines its precedence in receiving system resources.

private area. A major element of MVS/ESA virtual storage below the 16MB line. It contains the local system queue area (LSQA), scheduler work area, subpools 229 and 230, a 16KB system region area, and a private user region for running programs and storing data. This area is duplicated (except for the 16KB system region area) above the 16MB line as the **extended private area**.

profile. In CICS, a set of options specified in a resource definition that can be invoked by a transaction definition. Profiles control the interactions between the transaction and terminals or logical units. CICS supplies profile definitions suitable for most purposes. If a transaction definition does not specify a profile, a standard profile is used. In RACF, data that describes the significant characteristics of a user, a resource, a group of users, or a group of resources. See **resource profile**, **discrete profile**, **generic profile**, **user profile**, **resource group profile**, **data set profile**.

program check. A condition that occurs when programming errors are detected by a processor during execution.

program communication block (PCB). IMS control block that describes an application program's interface to an IMS database or, additionally, for message processing and batch message processing programs, to the source and destination of messages. See also **program specification block (PSB)**.

program compression. An operation performed by program control to relieve space in the DSA during a short-on-storage condition. The PPT is searched to identify programs that have been dynamically loaded and are currently not in use. If a program is not in use, the space it occupied is reclaimed.

program isolation (PI). An IMS facility that protects all activity of an application program from any other active application program until that application program indicates, by reaching a syncpoint, that the data it has modified is consistent and complete.

PRTYAGE. System initialization parameter used to specify the number of milliseconds to be used in the priority aging algorithm for incrementing the priority of a task. The default is 32 768 milliseconds. See the *CICS System Definition Guide* for more information.

PRVMOD. System initialization parameter used to specify the names of those modules that are not to be used from the LPA. See the *CICS System Definition Guide* for more information.

PSB directory (PDIR). A list or directory of program specification blocks (PSBs) that define for DL/I the use of data bases by application programs. It contains one entry for each PSB to be used during CICS execution, and is loaded during initialization. Each entry contains the size of the control block, the status, the storage location (if in storage), and the DASD address of the PSB in the ACBLIB. It is generated using DFHDLPSB macros. Contains entries defining each PSB to be accessed using local DL/I. Also contains entries for remote PSBs, to which requests are function-shipped using remote DL/I.

PSBCHK. System initialization parameter used to request DL/I security checking of a remote terminal initiating a transaction with transaction routing. This parameter is only applicable if the local CICS-DL/I interface is being used. The default is to have the remote link checked but no check made against the remote terminal.

PSBPL. System initialization parameter used to specify the size of the PSB pool in 1024-byte blocks for local CICS-DL/I interface support. This parameter is only applicable if the local CICS-DL/I interface is being used. The default is four blocks.

pseudoconversational. A type of CICS application design that appears to the user as a continuous conversation, but that consists internally of multiple tasks—also called "transaction-oriented programming."

purge. The abending of a task by task control to alleviate a short-on-storage condition.

PUT. Program update tape.

PVDELAY. System initialization parameter used define how long entries can remain in the PV signed-on-from list on the remote system. The default is 30 minutes. See **persistent verification**. See the *CICS System Definition Guide* for more information.

Q

QSAM. Queued sequential access method.

quasi-reentrant. Applied to a CICS application program that is serially reusable between entry and exit points because it does not modify itself or store data within itself between calls on CICS facilities.

queue. A line or list formed by items in a system waiting for service; for example, tasks to be performed, or messages to be transmitted in a message-switching system. In CICS, the transient data and temporary

storage facilities store data in queues. See **temporary storage (TS)**, **transient data (TD)**.

queued sequential access method (QSAM). An extended version of BSAM that incorporates queues of input and output blocks that are awaiting processing and transfer respectively.

R

RACF. The Resource Access Control Facility program product. An external security management facility available under MVS.

RAIA. Receive-any input area.

RAMAX. System initialization parameter used to specify the size in bytes of the I/O area allocated for each RECEIVE ANY issued by CICS. The default is 256 bytes. See the *CICS System Definition Guide* for more information.

RAPOOL. System initialization parameter used to determine the size of the CICS receive-any pool, which is set aside for VTAM receive-any operations. See the *CICS System Definition Guide* for more information.

real storage. The main storage in a virtual storage system. Physically, real storage and main storage are identical. Conceptually, however, real storage represents only part of the range of addresses available to the user of a virtual storage system.

receive-any control element (RACE). Type of control field held in the CICS receive-any pool set aside for VTAM receive-any operations. The number of RACEs maintained depends on the RAPOOL and MXT system initialization parameters and on the number of active tasks. See the *CICS System Definition Guide* for more information.

receive-any input area (RAIA). Type of input area held in the CICS receive-any pool set aside for VTAM receive-any operations. The number of RACEs maintained depends on the RAPOOL and MXT system initialization parameters and on the number of active tasks. See the *CICS System Definition Guide* for more information.

recoverability. The ability of a system to continue processing without loss of data when an unplanned interruption occurs.

recoverable in-doubt structure (RIS). In DBCTL, an area constructed for each unit of recovery when a failure occurs. Each RIS is written to the IMS log. RIS contents include the recovery token, the changed data records, and the identity of the data block that cannot be accessed because of unresolved in-doubts.

recovery. The process of returning the system to a state from which operation can be resumed. The restoration of resources following an error.

recovery manager. CICS resource recovery mechanism that provides a CICS resource manager, for example file control, with more flexibility than the DWE two-phase commit support for syncpoint and backout processing.

reference set. The amount of real storage required so that minimal (almost zero) virtual paging occurs. It is the total amount of real storage required to process the most frequently used sequence of instructions and data for a given set of transactions performing defined tasks, without causing any virtual storage paging operations.

region. A section of the dynamic area that is allocated to a job step or system task. The term is used to cover partitions and address spaces in addition to regions.

region-remote. A term used in early releases of CICS to refer to a CICS system in another region of the same processor. It can be taken to refer to a system that is accessed through an IRC (MRO) link, rather than through an SNA LU6.1 or LU6.2 link.

reliability. A measurement of the ability of a system to continue processing without failure. Shutting down an on-line system to process batch updates to the database subtracts from its availability to end users, but this has no bearing on reliability of components required to deliver the online service.

remote. In data communication, pertaining to devices that are connected to a data processing system through a data link. Synonym of link-attached. Contrast with **local**.

remote DL/I. A special case of function shipping, in which CICS sends a DL/I request to another CICS system. See also **function shipping**.

remote resource. In CICS intercommunication, a resource that is owned by a remote system. Contrast with **local resource**.

remote system. In CICS intercommunication, a system that the local CICS system accesses via intersystem communication or multiregion operation. Contrast with **local system**.

RENTPGM. System initialization parameter used to specify whether CICS is to allocate the extended read-only dynamic storage area (ERDSA) from read-only, key 0 protected storage (the default), or from CICS-key storage. Specifying CICS-key storage effectively creates a second ERDSA and allows ERDSA-eligible programs that execute in CICS-key to modify storage where required.

request parameter list (RPL). In ACF/VTAM, a control block that contains the parameters necessary for

processing a request for data transfer, for connecting or disconnecting a terminal, or for some other operation.

requested reset statistics. CICS statistics that the user has asked for by using the appropriate EXEC CICS or CEMT commands, which cause the statistics to be written to the SMF data set immediately. Requested reset statistics differ from requested statistics in that the statistics counters are reset, using an EXEC CICS or CEMT command.

requested statistics. CICS statistics that the user has asked for by using the appropriate EXEC CICS or CEMT commands, which cause the statistics to be written to the SMF data set immediately, instead of waiting for the current interval to expire. Contrast with **requested statistics**.

residence mode (RMODE). Attribute of a program indicating where it can reside, that is, either above or below the 16MB line.

resource. Any facility of the computing system or operating system required by a job or task, and including main storage, input/output devices, the processing unit, data sets, and control or processing programs.

Resource Access Control Facility (RACF). An IBM licensed product that provides for access control by identifying and verifying users to the system, authorizing access to protected resources, logging detected unauthorized attempts to enter the system, and logging detected accesses to protected resources.

resource control table (RCT). A control table that defines the relationship between CICS transactions and DB2 resources. For details, refer to the *DB2 Version 2 Administration Guide*.

resource measurement facility (RMF). An IBM program that collects system-wide data describing the processor activity (WAIT time), I/O activity (channel and device utilization), main storage activity (demand and swap paging statistics), and system resources manager (SRM) activity (workload). RMF produces two types of report, system-wide reports and address-space reports.

response time. The elapsed time from entry of a last input message segment to the first response segment.

restart. Resumption of operation after recovery. Ability to restart requires knowledge of where to start and ability to start at that point.

restart data set (RDS). A VSAM KSDS used only during emergency restart. The RDS temporarily holds the backout information read from the CICS system log. This allows CICS to be restored to a stable state and to be restarted following an abrupt termination.

RMODE. In MVS, an attribute that specifies the residence mode of the load module produced by the linkage editor. Unless a program is link-edited with RMODE(24), CICS loads it above the 16MB line if possible.

RMTRAN. System initialization parameter used with XRF to specify the name of the transaction that you want an alternate CICS to initiate when logged-on class 1 terminals are switched following a takeover. This parameter is applicable only on an alternate CICS region.

rollback. A programmed return to a prior checkpoint. In CICS, the cancellation by an application program of the changes it has made to all recoverable resources during the current logical unit of work.

rotational position sensing (RPS). A feature that permits a disk storage device to disconnect from a block multiplexer channel (or its equivalent), allowing the channel to service other devices on the channel during positional delay.

RPL. See **request parameter list**.

RPS. See **rotational position sensing**.

RSD (restart data set). The direct-access data set used to contain the information necessary to restart CICS.

S

SAM. Sequential access method.

sample statistics program (DFH0STAT). IBM-supplied batch program that provides information that is useful in calculating the storage requirements of a CICS system, for example, the sizes of the dynamic storage areas.

SAS (single address space). Single CICS region; usually used when contrasting with MRO.

scheduler work area (SWA). An element of the CICS address space. The SWA is made up of subpools 236 and 237 which contain information about the job and the step itself. Almost anything that appears in the job stream for the step creates some kind of control block in this area.

SCS. SNA character string.

SCS. System initialization parameter used to specify how much of the dynamic storage area (DSA) you want CICS to regard as the DSA storage cushion. The default is 64KB.

SDB. Structured database.

SDF. Screen Definition Facility. An online application development program product used to define or edit BMS maps interactively.

security. Prevention of access to or use of data or programs without authorization.

segment search argument (SSA). Segment search arguments (SSAs) are used to identify segments of a DL/I database. SSAs may be simple segment names or they may be qualified to include constraints made upon the values of fields within the named segment types. Except for a read-only operation, when it is unnecessary, SSAs used by a CICS application program must be in dynamic storage because of the requirement for the program to be quasi-reenterable.

sequential access method (SAM). An access method for storing and retrieving data blocks in a continuous sequence. Queued sequential access method (QSAM) extends the basic sequential access method (BSAM) by queuing the input and output blocks.

sequential data set. A data set whose records are organized on the basis of their successive physical positions, such as on magnetic tape.

service elements. The discrete hardware and software products that provide a terminal user with processing ability.

session recovery. The process in which CICS switches active sessions on class 1 terminals to backup sessions or reestablishes service on class 2 terminals.

short-on-storage (SOS). The condition in CICS that occurs when requests for storage from the dynamic storage areas exceed available storage. CICS cannot satisfy these requests, or can satisfy them only by using some of the storage cushions, even when all programs that are eligible for deletion, and are not in use, have been deleted. See also **storage cushion** and **program compression**.

single threading. The execution of a program to completion. Processing of one transaction is completed before another transaction is started. (Compare this with **multithreading**.)

SIP. CICS system initialization program.

SIT. System initialization parameter used to specify the suffix, if any, of the system initialization table (SIT) that you want CICS to load at the start of initialization. If you omit this parameter, CICS loads the pregenerated, default SIT, DFHSIT$$.

SMF. System management facilities.

SMF header. Describes the system creating the output.

SMF product section. Component of a CICS monitoring SMF record. It identifies the subsystem to which the monitoring data relates, which, in the case of CICS monitoring and statistics, is the CICS region.

SNA. Systems Network Architecture.

SNT. Signon table.

SOS. Short-on-storage.

SOT. CICS start of task.

SQL/DS. Structured Query Language/Data System. A relational database management facility.

SRB. Service request block (MVS).

SRL. System reference library—IBM-provided manuals that describe a programming or hardware product.

SRM. System resources manager—a component of the MVS control program.

SRT. System recovery table. System initialization parameter used to specify the system recovery table suffix. See the *CICS System Definition Guide* for more information.

SSA. Segment search argument.

startup. The operation of starting up CICS by the system operator.

startup jobstream. A set of job control statements used to initialize CICS.

statistics. System statistics are accumulated continually by CICS management programs in CICS system tables during the execution of CICS. System statistics can be captured and recorded, either on request or automatically at intervals, by any operator whose security code allows access to such information. In addition, system statistics are recorded on normal termination of the system.

See **automatic statistics**, **unsolicited statistics**, **end-of-day statistics**, **requested statistics**, and **requested reset statistics**.

statistics utility program (DFHSTUP STUP). Provides a summary report facility that can be used to interpret CICS statistics.

STATRCD. System initialization parameter used to set the statistics recording status at CICS initialization. The default is OFF; CICS interval and unsolicited statistics are not collected. End-of-day statistics are collected at the logical end of day and on shutdown. See the *CICS System Definition Guide* for more information.

STGPROT. System initialization parameter used to specify that storage protection is required in the CICS region. The default is NO: CICS runs in a single storage key. See the *CICS System Definition Guide* for more information.

STGRCVY. System initialization parameter used to indicate whether CICS should try to recover from a storage violation. The default is NO. See the *CICS System Definition Guide* for more information.

storage. A functional unit into which data can be placed and from which it can be retrieved. See **main storage, storage, virtual storage.**

storage accounting area (SAA). A field at the start of a CICS storage area that describes the area and enables CICS to detect some storage violations. Each CICS storage area has either an SAA or a storage check zone.

storage control. The CICS element that gets working storage areas.

storage cushion. A noncontiguous area of storage in the dynamic storage areas reserved for use by CICS when processing a short-on-storage condition.

storage dump. See **transaction dump**.

storage key. An indicator associated with each 4KB block of storage that is available in the CICS region. Access to the storage is controlled by **key-controlled storage protection**.

storage protection. A optional facility in CICS/ESA 3.3 that enables users to protect CICS code and control blocks from being overwritten inadvertently by application programs.

storage protection key. An indicator that appears in the current program status word whenever an associated task has control of the system. This indicator must match the storage keys of all main storage blocks that the task is to use.

storage violation. An error in a storage accounting chain in the dynamic storage area. A storage violation can be detected by the storage manager domain.

storage violation dump. A formatted dump taken as a result of a storage error detected by the storage control program, including a dump of the dynamic storage error.

subpool 229. An element of the CICS address space used primarily for the staging of messages. JES uses this area for messages to be printed on the system log and JCL messages as well as SYSIN/SYSOUT buffers.

subpool 230. An element of the CICS address space used by VTAM for inbound message assembly for segmented messages. Data management keeps data extent blocks (DEBs) here for any opened data set.

SUBTSKS. System initialization parameter used to define the number of task control blocks (TCBs) you want CICS to use for running tasks in concurrent mode. A concurrent mode TCB allows CICS to perform management functions such as system subtasks. The default is none. See the *CICS System Definition Guide* for more information.

summary report. A statistics report produced by the CICS statistics utility program (STUP). It summarizes the interval, unsolicited, requested reset, and end-of-day statistics on an applid by applid basis.

surveillance. In XRF, a series of processes by which the alternate CICS system monitors the active CICS system for a lapse of activity in order to detect potential failure conditions requiring a takeover. The active and alternate CICS systems use the CAVM surveillance mechanism to monitor each other's well-being.

surveillance signal. In XRF, the signal continuously written to the CAVM data sets by the active and alternate CICS systems to inform the each other of their states.

SVC. Supervisor call.

switch data traffic (SWDT). In an XRF configuration, a VTAM session control request sent to the NCP that initiates the switch of LU sessions from backup XRF session status to active XRF session status. The former XRF session, if still 'active', is terminated with an UNBIND. The switch request is issued to VTAM from the application program (alternate CICS system). VTAM passes the request to the boundary network node, where the sessions are actually switched by NCP.

switched connection. A connection that is established by dialing.

synchronization. The stage of the XRF process when the active and the alternate are both initialized, are aware of each other's presence, and the alternate is ready to begin tracking.

synchronization level (sync level). The level of synchronization (0, 1, or 2) established for an APPC session between intercommunicating CICS transactions. Level 0 gives no synchronization support, level 2 allows the exchange of private synchronization requests, and level 2 gives full CICS synchronization support with backout of all updates to recoverable resources if failure occurs.

synchronization phase. The XRF phase, immediately after initialization, when the alternate system builds the CICS control blocks to mirror those in the active system.

syncpoint. A logical point in execution of an application program where the changes made to the databases by the program are consistent and complete and can be committed to the database. The output, which has been held up to that point, is sent to its destination(s), the input is removed from the message queues, and the database updates are made available to other applications. When a program terminates abnormally, CICS recovery and restart facilities do not backout updates ***prior*** to the last completed syncpoint.

A syncpoint is created by any of the following:
- A DL/I CHECKPOINT command or CHKP call

- A DL/I TERMINATE command or TERM call
- An EXEC CICS SYNCPOINT request
- An end of task or an end of program.

See also **unit of work (UOW)**.

SYSEVENT data. A class of monitoring data which provides a special kind of transaction timing information.

SYSGEN. System generation.

system. In CICS, an assembly of hardware and software capable of providing the facilities of CICS for a particular installation.

system activity keypoint. A keypoint written to the system log automatically while CICS is running normally. (See also **activity keypoint**.)

system dump (SDUMP). In CICS, an MVS SDUMP, which may be formatted with a CICS-supplied IPCS exit to show all control blocks and storage areas in the CICS region.

system dump code. A code that identifies a user-defined entry in the system dump table. Each entry defines system actions and dump attributes to be associated with a code. Codes can be up to 8 characters in length.

A code consisting of the last six characters of a CICS message number describes actions to be taken and the dump to be produced when that message is issued. For example, a dump code table entry for ZZxxxx describes the system dump to be produced with message DFHZZxxxx, and overrides the documented system action for DFHZZxxxx.

Every system dump code can be invoked by EXEC CICS PERFORM DUMP SYSTEM commands and the corresponding CEMT transactions. For the definition of system dump code entries, see **system dump table**.

system generation (SYSGEN). In CICS, the process of creating a particular system tailored to the requirements of a data processing installation.

system initialization table (SIT DFHSIT). A CICS table that contains information to initialize and control system functions, module suffixes for selection of user-specified versions of CICS modules and tables, and information used to control the initialization process. You can generate several SITs, using the resource definition macro DFHSIT, and then use the SIT system initialization parameter to select the one that best meets your current requirements at initialization time.

system log. The (only) journal (identification='01') that is used by CICS to log changes made to resources for the purpose of backout on emergency restart.

system management facility (SMF). MVS management program. CICS stores monitoring and statistical data on SMF data sets.

system program. A program providing services in general support of the running of a system.

system queue area (SQA). A major element of MVS/ESA virtual storage below the 16MB line. This storage area contains tables and queues relating to the entire system. Its contents are highly dependent on the configuration and job requirements at installation. The equivalent area above the 16MB line is the **extended system queue area (ESQA)**.

system recovery table (SRT). A table listing the ABEND or abnormal condition codes that CICS will intercept.

system support program. A program product that defines and generates an NCP and provides it with utility programs.

Systems Network Architecture (SNA). The description of the logical structure, formats, protocols, and operational sequences for transmitting information units through and controlling the configuration and operation of networks.

T

takeover. In XRF, the shift of the workload from the active to the alternate CICS system, and the switching of resources needed for this to happen.

takeover phase. In XRF, the replacement of the failing active CICS system by the alternate CICS system as the session partner of the CICS users.

takeover time. In XRF, the elapsed time between the occurrence of a failure, the completion of switching all terminals to the alternate CICS system, and the running of the first user transaction.

task. A unit of work for the processor; therefore the basic multiprogramming unit under the control program. Under CICS, the execution of a transaction for a particular user. Contrast with **transaction**.

task control. The CICS element that controls all CICS tasks.

task control block (TCB). An MVS control block. A TCB is created for each MVS task. Several TCBs are created for CICS management programs. All CICS application programs and non-reentrant CICS code run under a single quasi-reentrant TCB.

task switching. Overlapping of I/O operations and processing between several tasks.

TCA. Task control area.

TCAM. Telecommunications access method.

TCP. System initialization parameter used to include the pregenerated non-VTAM terminal control program, DFHTCP.

TCT. System initialization parameter used to indicate which terminal control table, if any, is to be loaded.

TCTTE. Terminal control table terminal entry.

TCTUA. Option of the ADDRESS command, used also to pass information between application programs, but only if the same terminal is associated with the application programs involved (which can be in different tasks). The pointer reference is set to the address of the TCTUA. If a TCTUA does not exist, the pointer reference is set to X'FF000000'. The data area contains the address of the TCTUA of the principal facility, not that for any alternate facility that may have been allocated.

TCTUAKEY. System initialization parameter used to specify the storage key for the TCTUAs if CICS is operating with storage protection. The default is user-key: a user program executing in any key can modify the TCTUA. See the *CICS System Definition Guide* for more information.

TCTUALOC. System initialization parameter used to include where terminal user areas (TCTUAs) are to be stored. The default is below the 16MB line. See the *CICS System Definition Guide* for more information.

TD. CICS transient data. System initialization parameter used to specify the number of VSAM buffers and strings to be used for intrapartition transient data. See the *CICS System Definition Guide* for more information.

Telecommunications Access Method (TCAM). An access method used to transfer data between main storage and remote or local storage.

Teleprocessing Network Simulator (TPNS). A program used to test new functions before they encounter production volumes.

temporary storage (TS). A CICS facility for temporarily saving data in the form of sequential queues. A TS queue is held in main storage or on a VSAM data set on DASD. All queues not in main storage are in a single VSAM data set. A task can create a TS queue with a name selected by the task. The queue exists until deleted by a task (usually, but not necessarily, the task that created it). Compare **transient data**. Possible uses of temporary storage include storage of screen images for terminal paging and storage of incomplete data for suspended tasks. In general, TS queues do not require resource definition, but see **temporary storage table (TST)**.

terminal. In CICS, a device, often equipped with a keyboard and some kind of display, capable of sending and receiving information over a communication channel. A point in a system or communication network at which data can either enter or leave.

terminal control. The CICS modules that control all CICS terminal activity.

terminal control program (TCP). The program that controls all CICS terminal activity.

terminal control table (TCT). CICS control table retained to define non-VTAM terminal networks.

terminal input/output area (TIOA). Area that is set up by storage control and chained to the terminal control table terminal entry (TCTTE) as needed for terminal input/output operations.

terminal list table (TLT). CICS control table that allows terminal, or operator identifications, or both, to be grouped logically. See **supervisory terminal functions**.

terminal paging. A set of commands for retrieving pages of an oversize output message in any order.

terminal-owning region (TOR). A CICS region which owns most or all of the terminals defined locally. See also **application-owning region (AOR)**, **data-owning region (DOR)**.

termination phase. The XRF phase in which the XRF complex returns to two separate and independent environments and all XRF activity in the alternate system stops.

thread. In CICS, a link between a CICS application and DBCTL. To DBCTL, a thread represents the CICS transaction that has issued a DL/I request. The system initialization parameter DLTHRED specifies the number of threads provided through the CICS local DL/I interface.

threading. The process whereby various transactions undergo concurrent execution.

throughput. The total data processing work successfully completed during an evaluation period.

throughput rate. The data processing work successfully completed per unit of time.

TIOA. Terminal input/output area. TIOAs are acquired and chained to the TCTTE as needed for terminal input/output operations. The field TCTTESC addresses the first terminal-class storage area obtained for a task (the beginning of the chain) and the field TCTTEDA gives the address of the active TIOA. CICS terminal control passes data received from a terminal to the CICS application program in the TIOA, and writes data from the TIOA to the terminal.

TLT. Terminal list table.

trace. Facility for recording CICS activity. There are three destinations for trace entries: internal trace, auxiliary trace, and generalized trace facility (GTF).

trace control. The CICS element that provides a trace facility.

tracking. In XRF, the process by which the alternate CICS system mirrors the starting and stopping of terminal sessions in the active CICS system so that it is prepared to take over the active system should the need arise.

transaction backout. The cancelation, as a result of a transaction failure, of all updates performed by a task.

transaction dump. A dump of the control blocks and storage areas associated with a particular task.

transaction identification code. Synonym for transaction identifier. For example, a group of up to four characters entered by an operator when selecting a transaction.

transaction identifier. A name of up to four characters that is specified when the transaction is defined to CICS and and that is used to invoke the transaction. For example, to select a transaction, a terminal operator enters the transaction identifier.

transaction rate. The number of units of processing successfully completed per unit of time.

transaction restart. The restart of a task after a transaction backout.

transient data (TD). A CICS facility for temporarily saving data in the form of queues, called destinations. A TD destination is held as either as a queue in a VSAM data set managed by CICS (intrapartition TD) or as a QSAM data set outside the CICS region. See **intrapartition transient data** and **extrapartition transient data**. Contrast with **temporary storage**.

TRTABSZ. System initialization parameter used to specify the size, in kilobytes, of the internal trace table. The default (and minimum) size is 16KB.

TST. System initialization parameter used to specify the suffix of the temporary storage table. See the *CICS System Definition Guide* for more information.

tuning. The process of adjusting system control variables to make the system divide its resources most efficiently for the workload.

turnaround time. The elapsed time between entry of the first character of the first input into the input interface and the passage of the last character of the last output through the output interface. The total time consumed from the start to the completion of a specific unit of work measured at specific interfaces. When multiple inputs or multiple outputs are parts of one unit of work, intermediate turnaround time specifications may be needed.

TWA. Transaction work area. Option of the ADDRESS command, used also to pass information between application programs, but only if they are in the same task. The pointer reference is set to the address of the TWA. If a TWA does not exist, the pointer reference is set to X'FF000000'.

TWX. Teletypewriter exchange terminal.

U

unsolicited statistics. CICS statistics automatically gathered by CICS for dynamically allocated and deallocated resources. See also **interval statistics**, **end-of-day statistics**, **requested statistics**, and **requested reset statistics**.

unit of work (UOW). A sequence of processing actions (database changes, for example) that must be completed before any of the individual actions can be regarded as committed. When changes are committed (by successful completion of the UOW and recording of the syncpoint on the system log), they do not need to be backed out after a subsequent failure of the task or system. The end of a UOW is marked in a transaction by a syncpoint, issued either by the user program or by CICS at the end of task. In the absence of user syncpoints, the entire task is a UOW.

unwanted takeover. In XRF, a takeover initiated by the alternate CICS system when there was not an actual failure on the active CICS system. This might be due to an unusual system condition which, although not a true failure, slowed down the active system's participation in the surveillance process to the point where the alternate system believed that a failure on the active system had occurred.

update. To modify a file or data set with current information.

user activity keypoint. A keypoint written to the system log by a user transaction. See also **activity keypoint**.

user dynamic storage area (UDSA). A storage area in CICS/ESA 3.3 allocated below the 16MB line and reserved exclusively for those user application programs that execute in user-key and that reside below the 16MB line.

user exit. A point in an IBM-supplied program at which a user exit routine may be given control. For programming information, see the *CICS Customization Guide*.

user-key. Storage obtained by CICS in MVS open-key storage. It is for user application programs and their

associated data areas. It can be accessed and modified by user applications and by CICS. See **CICS-key**, **storage protection**.

V

validity of reference. Direct reference to the required pages, without intermediate storage references that retrieve unwanted data.

virtual machine (VM). A functional simulation of a computer and its associated devices. Contrast with **real machine**.

virtual storage (VS). (ISO) The notional storage space that may be regarded as addressable main storage by the user of a computer system in which virtual addresses are mapped into real addresses. The size of virtual storage is limited by the addressing scheme of the computing system and by the amount of auxiliary storage available and not by the number of main storage locations.

Virtual Storage Access Method (VSAM). An access method for direct or sequential processing of fixed- and variable-length records on direct access devices.

virtual storage constraint relief (VSCR). The movement of areas of code or control blocks to storage above the 16MB line, or the reduction of code or control blocks below the 16MB line. These actions increase the storage available for user programs and data that use 24-bit addressing.

virtual storage paging. A technique used by CICS in a virtual storage environment. The key objective of programming in this environment is the reduction of page faults. A page fault occurs when a program refers to instructions or data that do not reside in real storage, in which case, the page in virtual storage that contains the referenced instructions or data must be paged into real storage. The more paging required, the lower the overall system performance.

VLF. The Virtual Lookaside Facility in MVS/ESA. Manages the data space associated with library lookaside (LLA).

VS. See **virtual storage**.

VSAM shared resources. Buffers and strings shared by several VSAM data files. This is defined to CICS in the file control table.

VSAM work area (VSWA). An area that is acquired dynamically by the file control program when accessing a VSAM data set.

VSCR. Virtual storage constraint relief.

VTAM. Virtual telecommunications access method. System initialization parameter used to include the VTAM access method. The default is YES. See the *CICS System Definition Guide* for more information.

VTAM Performance Analysis and Reporting System (VTAMPARS). A program offering that provides information on network traffic through the VTAM component of the network.

W

warm keypoint. A keypoint written to the restart data set during controlled shutdown (after all system activity has ceased). During a subsequent warm restart, information in the warm keypoint is used to reestablish system tables to the status they had at controlled shutdown. See also **keypoint**.

warm start. Initialization of a CICS system using selected system status information obtained during the previous termination. Automatic restart after a normal (controlled) shutdown.

working set. The set of a user's pages that must be active in order to avoid excessive paging. The amount of real storage required in order to avoid excessive paging.

X

XLT. Transaction list table. CICS control table containing a list of logically-related transaction identifications. System initialization parameter used to specify the suffix for the transaction list table. See the *CICS System Definition Guide* for more information.

XMOLS. CICS monitor listing program, DFH$MOLS.

XRF (Extended Recovery Facility). Extended Recovery Facility (XRF).

XRF complex. The MVS images and licensed programs that provide an XRF service.

Index

Numerics

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

R

S

T

U

V

W

X

Sending your comments to IBM

If you especially like or dislike anything about this book, please use one of the methods listed below to send your comments to IBM.

Feel free to comment on what you regard as specific errors or omissions, and on the accuracy, organization, subject matter, or completeness of this book.

Please limit your comments to the information in this book and the way in which the information is presented.

To request additional publications, or to ask questions or make comments about the functions of IBM products or systems, you should talk to your IBM representative or to your IBM authorized remarketer.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate, without incurring any obligation to you.

You can send your comments to IBM in any of the following ways:

- By mail, to this address:

  Information Development Department (MP095)
  IBM United Kingdom Laboratories
  Hursley Park
  WINCHESTER,
  Hampshire
  SO21 2JN
  United Kingdom
- By fax:
  - From outside the U.K., after your international access code use 44–1962–870229
  - From within the U.K., use 01962–870229
- Electronically, use the appropriate network ID:
  - IBM Mail Exchange: GBIBM2Q9 at IBMMAIL
  - IBMLink™: HURSLEY(IDRCF)
  - Internet: idrcf@hursley.ibm.com

Whichever you use, ensure that you include:

- The publication number and title
- The topic to which your comment applies
- Your name and address/telephone number/fax number/network ID.

Program Number: 5655-147

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.

SC33-1699-33

Spine information:

CICS TS for OS/390 CICS Performance Guide Release 3