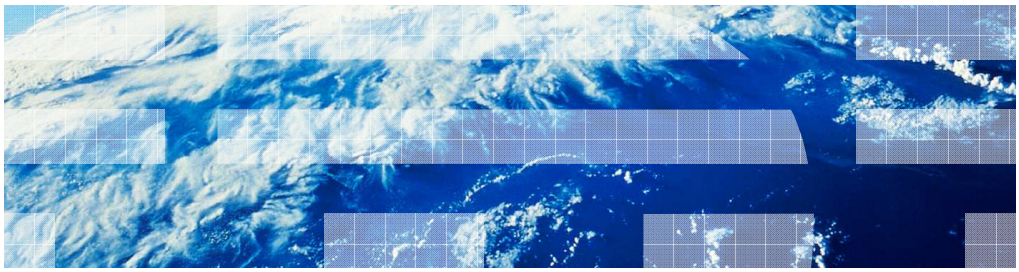
# CICS Transaction Server V4.2

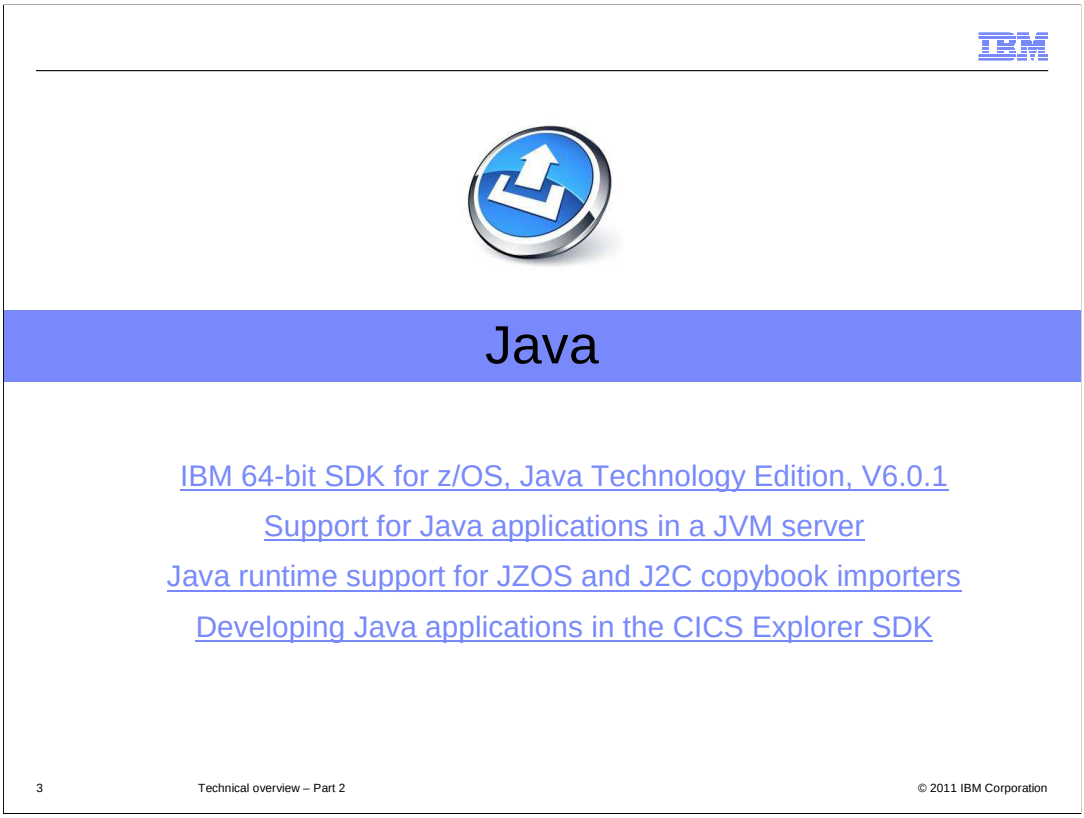
## Technical overview – Part 2





This module is part 2 of the technical overview of CICS® Transaction Server version 4.2.


The enhancements to CICS TS 4.2 are split into five themes to deliver a smart transaction processing experience: Events, Connectivity, Java, Management, and Scalability. This module covers the enhancements that fall into themes of Java, management, and scalability. Part one of the technical overview describes the product strategy, summarizes the key themes, and goes into more detail on two of those themes: event processing and connectivity.




This release has significant enhancements for running Java applications in CICS. CICS uses the 64-bit version of the Java SDK. Version 6.0.1 provides performance improvements for running Java on z/OS® and is the prerequisite for the CICS TS 4.2 release. The JVM server runtime environment provides the ability to run many Java applications in a single JVM. You can use the JZOS and J2C copybook importers to create Java classes from copybooks. The CICS Explorer™ SDK provides an environment for developing and deploying CICS Java applications to any in service release of CICS. These enhancements are summarized here. Select any of the links on this slide to read more in the CICS Information Center.

## IBM 64-bit SDK for z/OS, Java Technology Edition, V6.0.1

- **64-bit provides significant more storage**
  - Removes significant constraint and complexity in managing large Java workloads
  - Applies to JVM servers and existing pooled JVMs
    - Java stack and heap are now allocated in above-the-bar storage
    - Allows more Pooled JVMs per region
    - Allows more storage to increase caches and reduce frequency of GC
  - No Java application changes
    - Requires code using the JNI to be 64-bit – typically only used by vendors
    - OSGi bundled 64-bit drivers available with DB2® and WebSphere® MQ
- **V6.0.1 is IBM zEnterprise™ optimized for significant performance gains**
  - Exploits new z196 / z114 instruction set
  - Improved garbage collection and just in time compiler
  - Eligible for offload to zAAP
- **Removed support previous and 31-bit JVM versions**

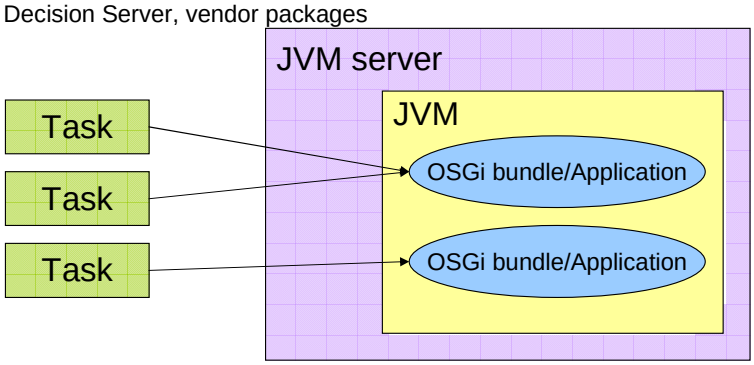



Support for the 64-bit SDK enables you to run Java applications in CICS without effecting the storage of existing 31-bit applications, and significantly increases the amount of storage available to Java applications. The SDK can be used with JVM servers and existing pooled JVMs. The main changes are that the Java stack and heap are allocated in 64-bit storage, allowing many more pooled JVMs to run in each region. It also means you can run with larger caches and reduce the frequency of garbage collection. You do not have to change your Java applications, but you must ensure that any code using the Java Native Interface can run in 64-bit. 64-bit drivers are available to access DB2 and WebSphere MQ. See the CICS system requirements for details.

Version 6.0.1 is optimized to run on System z®. It exploits the new instruction sets, has improved garbage collection and just-in-time compiler, and work is eligible to run on a zAAP. You cannot run previous versions of the SDK and you must use the 64-bit version of 6.0.1. The 31-bit version is not supported by CICS.

## Support for Java applications in a JVM server

- **JVMSERVER is a new CICS resource that defines a long-running JVM instance**
  - Serves multiple transactions concurrently
  - Application tasks run exclusively as OSGi bundles
  - New statistics available in CICS Explorer and monitoring records
- **Strategic direction of Java in CICS to host new workloads**
  - Axis2, Dynamic Scripting Feature pack V1.1 for CICS TS
  - CICS Transaction Server support for WebSphere Compute Grid - SupportPac CN11
  - IBM Decision Server, vendor packages



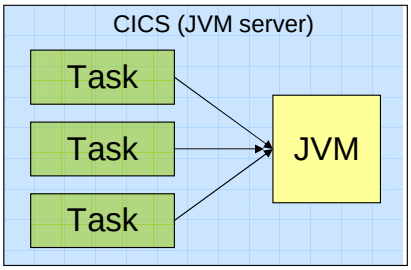


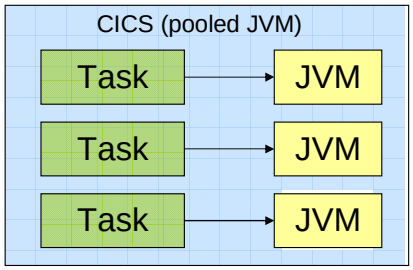


The JVM server is a runtime environment for running multiple CICS transactions concurrently in the same JVM. It is represented by the JVMSERVER resource. Applications can run in the same JVM server if they are threadsafe and packaged as OSGi bundles. The JVM server contains an OSGi framework to run the OSGi bundles. CICS produces statistics that you can view in CICS Explorer to manage and tune the environment for optimal performance.

The JVM server is the strategic direction for running Java in CICS, including new workloads such as Axis2 for web services and the Dynamic Scripting Feature Pack. The JVM server is also used for the WebSphere Compute Grid supportpac and can be extended to include IBM Decision Server and other external packages.

## JVM server versus existing Java support




Single JVM – serves many tasks (reduced storage)

Concurrent, multi-threaded, up to 256 threads per JVM server

T8 (CICS key)

MAXTHRDTCBS automatically calculated, up to maximum of 1024 per region

More standard server model (+ data sharing)

Dynamic update and replace of modules

Pool of JVMs – search serves only a single task

Java program isolation

J8 (CICS key), J9 (User key)

MAXJVMTCBS system initialization parameter

Difficult to share data and state

JVMs must be restarted to effect changes




The JVM server provides many advantages over the existing pooled JVM support. You can use a single JVM to run many tasks, reducing the required storage, as opposed to running many JVMs where each one serves only a single task. The JVM server provides concurrent, multithreaded support. You can have a maximum of 256 threads in each JVM server. Other differences include the TCBs that are used to run the work and the system initialization parameters that are required. One of the main advantages is that the JVM server is more like a standard server model that can perform data sharing, whereas in a pooled JVM it is difficult to share data and state. Another advantage is that you do not need to restart the JVM server to update and replace application modules because they are packaged as OSGi bundles. In a pooled environment, you have to restart the JVMs to pick up application changes.

## Support for OSGi Service Platform

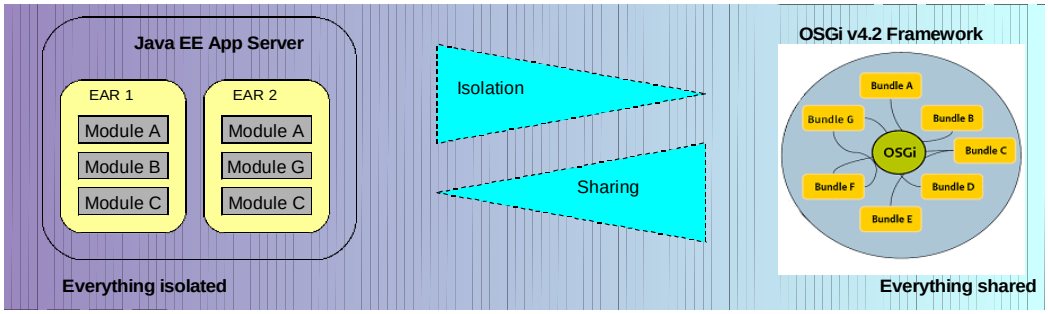
- **OSGi Service Platform**
  - Standards-based framework for deployment and management of Java applications
  - JVM server includes Equinox an implementation of the OSGi Service Platform
  - Provides application version and pre-req management, and isolation
- **Java applications are required to use OSGi development and packaging**
  - Existing CICS Java applications using main() method linkage can run unchanged if "wrapped" in an OSGi bundle
  - Java applications must be threadsafe and cannot use CICS EJB or CORBA
- **CICS BUNDLE resource provides deployment life cycle for OSGi bundles**
  - New copy of Java code using discard, then install of a BUNDLE resource
  - OSGi provides built in version and pre-req management
- **Dedicated class loader for each OSGi bundle**
  - Application isolation
  - Improved class validation – no more NoClassDefFoundError




The OSGi Service Platform provides a standard framework for packaging, deploying, and managing Java applications. The JVM server includes the Equinox implementation of the OSGi Service Platform. OSGi provides advanced application versions, dependency management, and isolation. To run a Java application in a JVM server, you must use OSGi development and packaging. However, you do not need to make changes to existing applications if they are threadsafe. You can wrap the existing application in an OSGi bundle. Also you cannot use enterprise Javabeans or CORBA in a JVM server.

The CICS BUNDLE resource provides lifecycle management of the OSGi bundles, so you can install, enable, disable, and discard a collection of related OSGi bundles together. The OSGi framework resolves dependencies based on the information in the OSGi bundle. Each OSGi bundle also has a dedicated class loader to ensure application isolation and improve class validation.

## OSGi – isolated and shared bundles




- In Java EE, modules are isolated within an application and applications are isolated from one another
  - Makes sharing modules difficult
- In OSGi, all bundles have shared visibility to the externals of all other bundles in an OSGi framework (JVM)
  - Bundles have to include import declarations to use other bundles in the framework




Unlike Java EE applications, OSGi based Java applications allow an extensive and flexible reuse of Java components. Any Java class loaded in an OSGi bundle can be shared to any other bundle residing in the same OSGi framework. In fact, while Java EE ensures isolation between EAR applications loading their classes on isolated class loaders, in the OSGi architecture bundle class loaders can share their classes. The application developer has to explicitly add import statements to use packages from other bundles at run time.

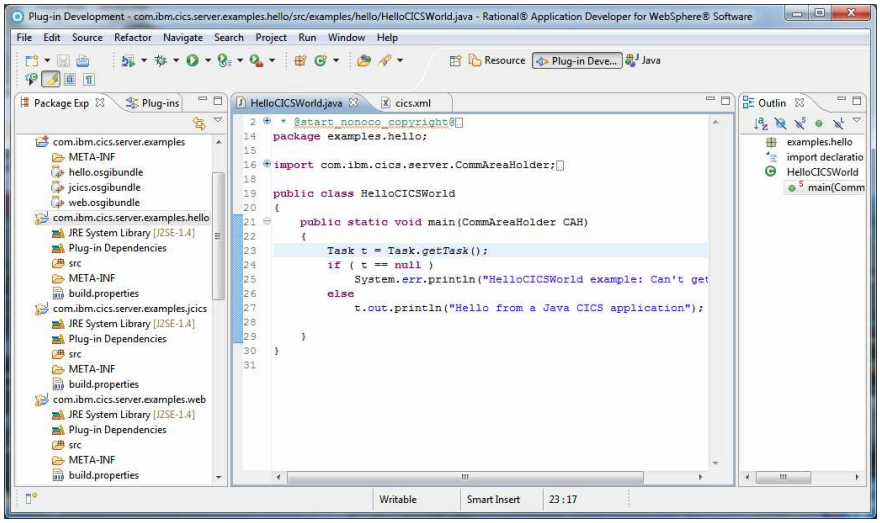
## Developing Java applications in the CICS Explorer SDK

- **CICS Explorer SDK**
  - Eclipse development toolkit for OSGi packaging and deployment to CICS
  - Requires Eclipse 3.6.2, or later
    - Freely available to download from http://www.eclipse.org
  - OR a product already based on Eclipse 3.6.2, such as IBM Rational® Application Developer V8.0
  - Supports deployment to all CICS releases
    - JVM server at CICS TS V4.2 only
    - JVM pool at CICS TS V3.2, and above
- **Java projects are developed as Plug-in Projects and then packaged in a CICS bundle and exported to zFS**




To help with developing and deploying OSGi bundles into CICS, the CICS Explorer SDK has been enhanced to provide this support. The CICS Explorer SDK provides application development capabilities and includes support for developing Java applications using the JCICS API and deploying into CICS. You must have an Eclipse Integrated Development Environment at Version 3.6.2 or later. The IDE is freely available from the Eclipse website. Alternatively, you can use a product that is based on Eclipse 3.6.2, such as IBM Rational Application Developer Version 8. The CICS Explorer SDK supports all in-service releases of CICS, so you can develop Java applications for any release. You develop Java projects as plug-in projects in the tool and then package the application in a CICS bundle and export to zFS.

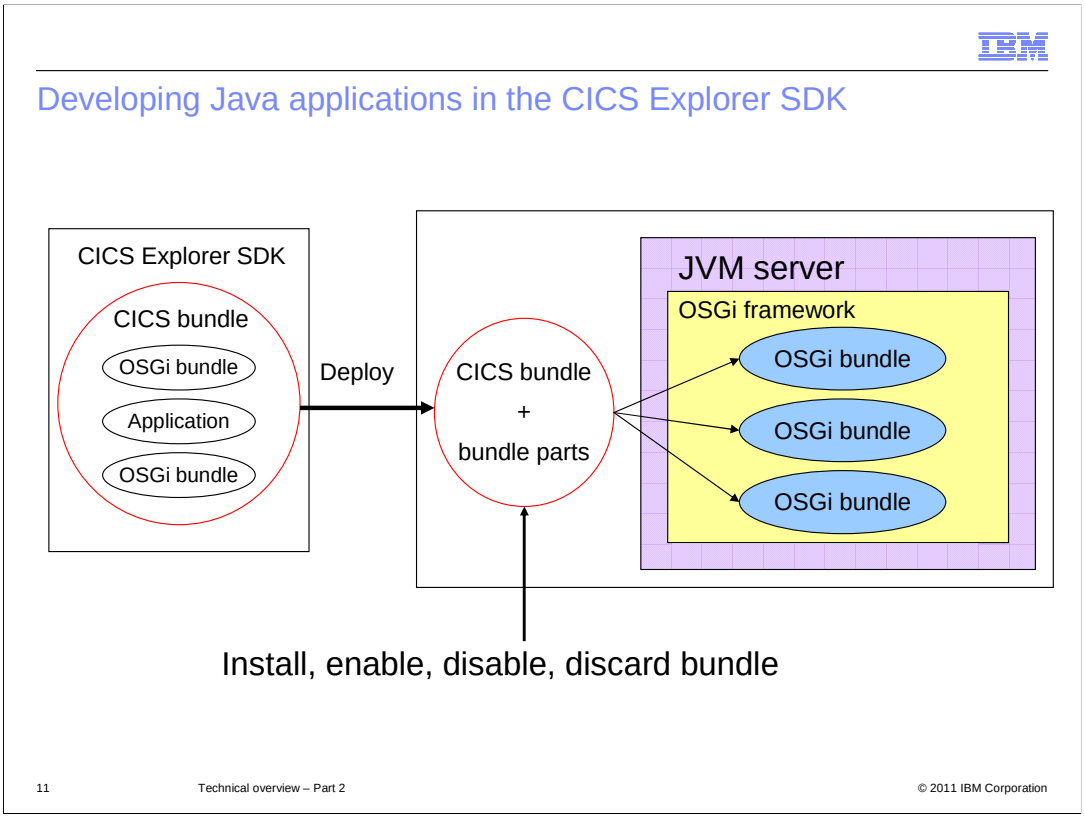
## CICS Explorer SDK – for Java development




- CICS Explorer SDK now provides complete toolkit for developing and deploying CICS Java applications




This screen capture shows the CICS Explorer SDK installed in Rational Application Developer with a plug-in project that uses one of the CICS Java samples.




In the CICS Explorer SDK you can create one or many OSGi bundles for an application. Use the CICS Bundle project wizard to create a CICS bundle to deploy the application to CICS. You can install, enable, disable, and discard the CICS BUNDLE resource to manage the life cycle of the application. The OSGi framework handles the dependencies between the OSGi bundles in the framework.

## Java runtime support for JZOS and J2C copybook importers

- **Tools and APIs to aid converting data types between COBOL and C/C++ language structures and Java objects**
  - Program interfaces – LINK, START, XCTL
  - Data – VSAM files, TS, TD
- **Data binding beans from the J2C tools and wizards included with**
  - IBM Rational Application Developer
  - IBM Rational Developer for System z
- **IBM JZOS Batch Toolkit for z/OS SDK**
  - A version of the JZOS Batch Toolkit is included as part of the IBM Java SDK for z/OS
  - http://www.alphaworks.ibm.com/tech/zosjavabatchtk
- **Runtimes for the products listed here have been tested in the CICS JVM server**




If you are developing Java applications that interface with COBOL, C, or C++ language structures, you can use tools and APIs to help. Both Rational Application Developer and Rational Developer for System z include data binding beans from the J2C tools and provide wizards in the tools to help you. In addition, a JZOS batch toolkit is provided with the IBM Java SDK for z/OS. The runtimes for both J2C and JZOS have been tested in a JVM server.

# Management

Transaction tracking support

Enhancements to main and auxiliary temporary storage queues

Improvements to dynamic workload management

WLM routing for IPIC requests

Password phrases

System availability monitoring

Dynamic transaction-level control for workload routing

CMCI support for task history recording…*and sorting records*

New WLM routing algorithms to exclude link weighting

MVS job ID and system initialization parameters




The management theme includes many enhancements to make it easier for system programmers and administrators to manage CICS and troubleshoot problems. These enhancements are summarized here. Select any of the links on this slide to read more in the CICS Information Center.

## Transaction tracking

- **Track a transaction through the system**
  - Modern transactions (web, MQ, CICS TG etc) do not have termids/sessions as principal facilities
  - IP addresses are not easily managed using traditional tools (CEMT etc)
    - Hard to track transactions across a web of associated regions
- **User transactions have *association data* consisting of:**
  - Origin data - created at *point of origin*
  - Task context data - created for every task
  - Previous hop data – created for remote requests
  - Adapter data - created by STARTs from TRUE based adapters
  - User correlation data - created by user exit
- **Origin data includes a unique tracking token for all *associated* tasks**
  - Transaction group ID (trngrpid)
  - Spans STARTs, web services pipeline, HTTP, CTG, DPL, FS, TR etc
  - Supported for MRO and IPIC connection
  - Not supported for SNA connections or DTP programs




Transaction tracking helps you follow transactions through a CICS system. Many transactions, for example those that originate from the web, WebSphere MQ, and CICS Transaction Gateway, do not have terminal IDs or sessions as principal facilities. IP addresses are not easily managed by some interfaces such as CEMT, so it can be hard to track transactions across many associated regions.

In CICS TS 4.2, user transactions have association data, including the point of origin, the task context, previous hop data for remote requests, and additional data from user exits. Origin data includes a unique tracking token for all associated tasks. It is supported for MRO and IPIC connections, but not for SNA or distributed transaction processing programs.

## Transaction tracking in V4.2

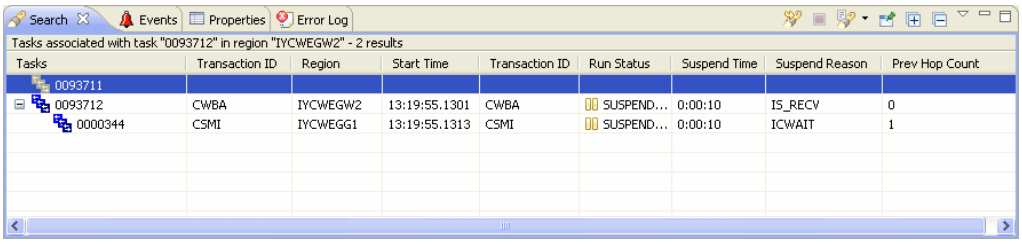
- **CICS TS 4.2 extends content and uses of association data**
  - All user tasks now create association data (CEMT, CEDA)
  - Transmission supported over MRO
  - Origin data section enhanced to include new adapter data
    - "Non-CICS" adapters can set origin data fields when initiating work by non-terminal STARTs
    - Exploited by MQ trigger monitor and WOLA V8.0.0.1
  - New **Previous Hop** data section
    - Describes the remote sender of the request so that a request can be tracked back to the initiating task in the previous CICS system



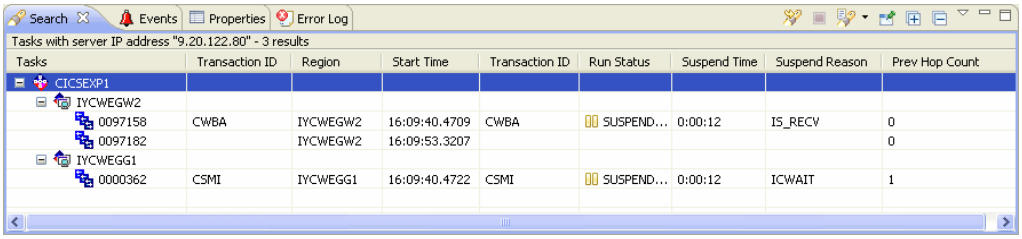

In CICS TS 4.2, all user tasks now create association data. Transmission supported over MRO. The origin data now includes new data when work is initiated from non-terminal start requests. This data is also exploited by the WebSphere MQ trigger monitor and WOLA. Additionally, the previous hop information includes data about the remote sender of the request, so a request can be tracked back to the initiating task in another CICS system.

## Transaction tracking – Explorer views

Associated tasks, with previous hop data




Tasks originating from an IP address

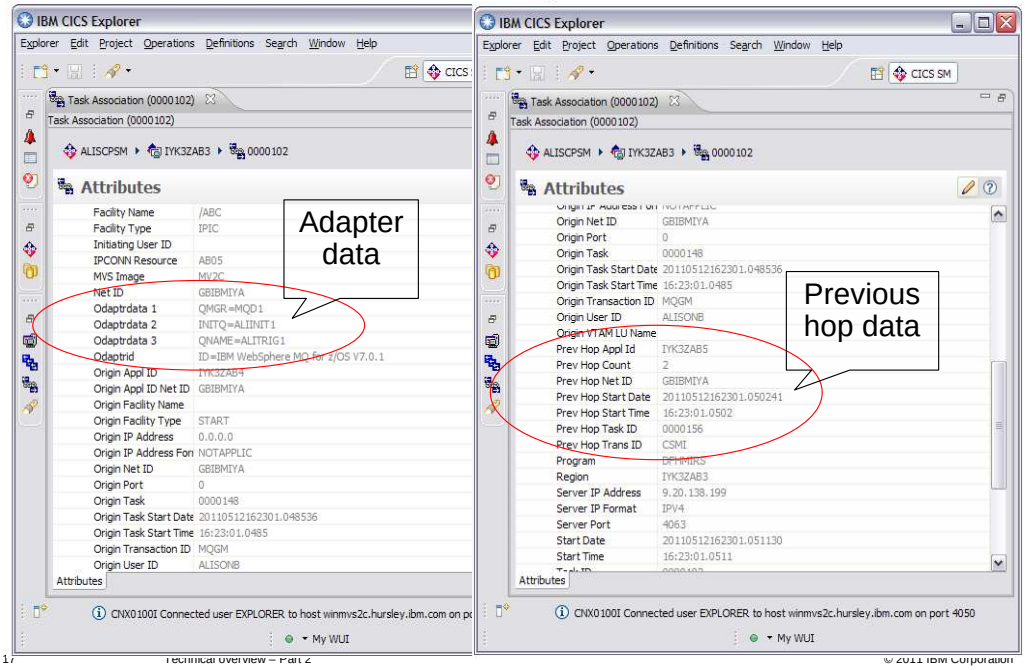




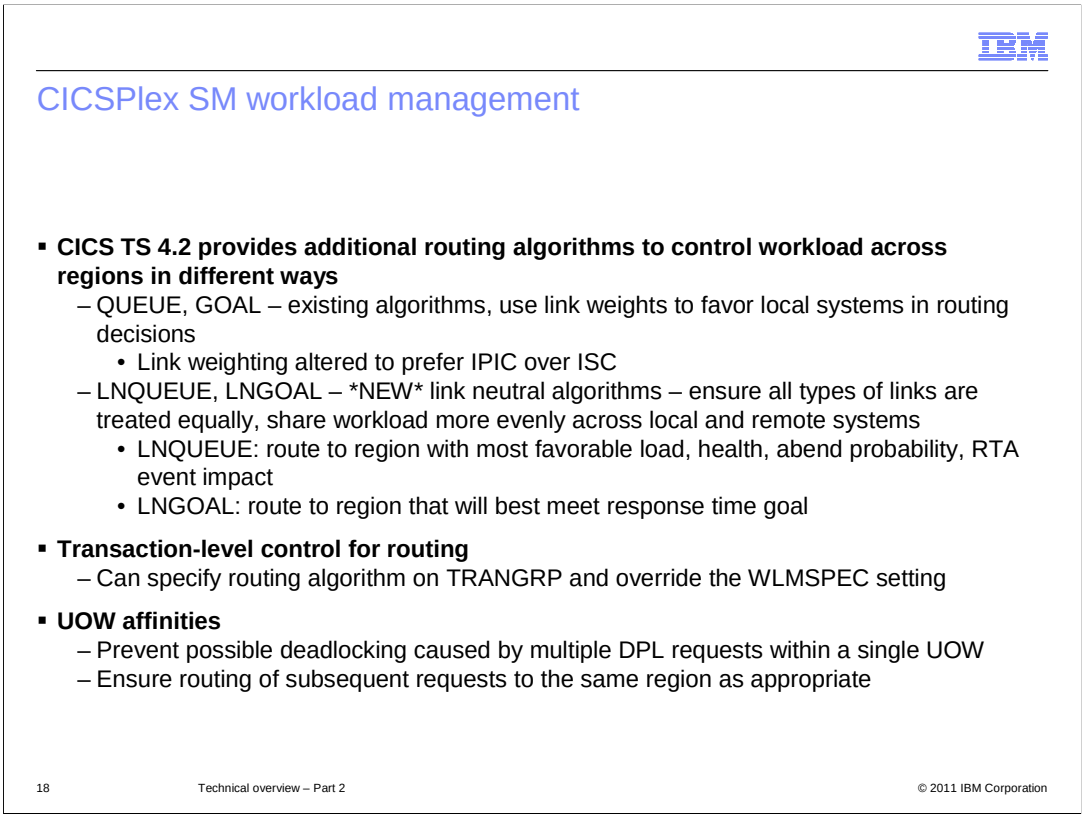


You can view transaction tracking information in CICS Explorer as shown in these two screen captures.

## Adapter data and previous hop data




The task association information has several fields for adapter data and previous hop data that can tell you where the task originated.




CICS provides two new routing algorithms to control workloads across regions. For the existing queue and goal algorithms, the link weighting has altered to prefer IP interconnectivity over intersystem communication. Link neutral queue and goal algorithms are new in this release. They ensure all types of links are treated equally, sharing the workload more evenly across local and remote systems. The link neutral queue algorithm routes work to the region with the most favorable load, health, abend probability, and RTA event impact. The link neutral goal algorithm routes work to the region that will best meet the response time goal. You can also control routing at the transaction level, by specifying a routing algorithm on the transaction group and overriding the workload specification. Unit of work affinities are also now handled to prevent the possible deadlock of multiple distributed programming link requests occurring in a single unit of work. CICSPlex® System Manager ensures that the subsequent requests are routed to the same region as appropriate.

## New CICSPlex SM SYSPARM resource

- **New SYSPARM base table has been added to CICSPlex SM**
- **Allows retrieval of systems initialization table values**
  - Read-only access to SIT parameters
  - Ability to access SIT parameters from
    - DFHSITxx load module
    - EXEC PGM=DFHSIP,'PARM=...'
    - SYSIN data set overrides
    - Console overrides
    - All of the access mechanisms combined together in order
  - Cannot retrieve the changes made after startup using system programming interface commands



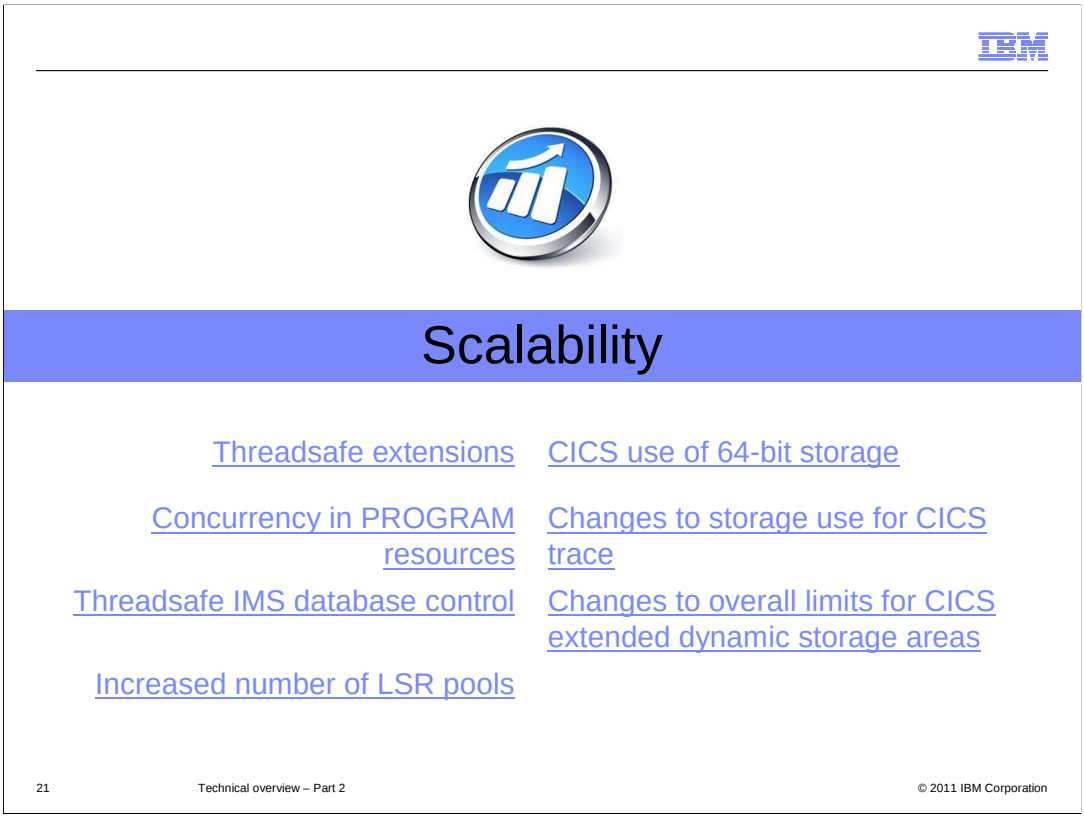

CICSPlex SM has a new resource table called SYSPARM. It allows retrieval of system initialization parameter values and overrides from the different places where they can be set in CICS. However, you cannot retrieve the changes made after startup using the SPI commands. You can access system initialization parameter values in CICS Explorer.

## Password phrase support

- **Password phrases are an alternative to traditional passwords**
  - Provide improved system security - harder to attack, easier to remember
  - Character string comprising mixed-case letters, numbers, and certain special characters, up to 100 characters in length
    - ESM specifies exact rules
    - Minimum length defined by RACF® (14 characters unless ICHPWX11 exit present and allows nine characters)
- **CICS introduced support for password phrases by most major CICS interfaces including:**
  - New sign-on transaction (CESL)
  - CICS API
  - Web
  - Web services
  - CMCI
  - WUI




This release of CICS TS provides support for password phrases, an alternative to traditional passwords. Password phrases provide improved security, making it harder to hack into the system. A password phrase is a character string that comprises mixed-case letters, numbers, and certain special characters up to a maximum of 100 characters in length. The security manager provides the exact rules for what is allowed. In RACF, the minimum for a password phrase is 14 characters unless an exit is present to allow a minimum of nine characters. Password phrases can be used by most CICS interfaces, including a new sign-on transaction.




The scalability theme focuses on making better use of hardware, lifting restrictions around CICS usage of the operating system, and providing more threadsafety to alleviate constraints on the QR TCB. These enhancements are summarized here. Select any of the links on this slide to read more in the CICS Information Center.

## Threadsafe extensions – threadsafe SYNCPOINT command

- **Commands now made threadsafe***
  - EXEC CICS SYNCPOINT
  - EXEC CICS SYNCPOINT ROLLBACK
  - EXEC CICS RESYNC*
  - Some switching might still occur, but heavily reduced
- **EXEC API no longer switches to QR for syncpoint requests**
  - Recovery Manager (RM) domain now makes the decision if a switch is required
  - All RM domain clients register at startup and tell RM if they are threadsafe
  - RM domain keeps it on the open TCB wherever possible
- **End of task sync point can run on an open TCB before CICS switches to QR to terminate the task**




Syncpoint and rollback processing, including SYNCPOINT and RESYNC commands, have been made threadsafe to avoid TCB switching where possible. These updates are also available for CICS TS 4.1 as APARs. The Recovery Manager domain now decides if a switch is required on a syncpoint request. Where possible the work runs on an open TCB. The end of task sync point can run on an open TCB before switching back to the QR to terminate the task.

## Other API commands made threadsafe

- QUERY SECURITY
- SIGNON, SIGNOFF
- VERIFY PASSWORD, VERIFY PHRASE
- CHANGE PASSWORD, CHANGE PHRASE
- EXTRACT TCPIP, EXTRACT CERTIFICATE
- All Call and EXEC Level Named Counter Server commands
- Built in functions for DIGEST and DEEDIT




Additional commands have been made threadsafe in CICS TS 4.2 and they are listed here.

## Threadsafe SPI commands

- **New SPI commands that are threadsafe:**
  - INQUIRE CAPDATAPRED, INQUIRE CAPINFOSRCE, INQUIRE CAPOPTPRED
  - INQUIRE EPADAPTER, SET EPADAPTER
  - INQUIRE OSGIBUNDLE, INQUIRE OSGISERVICE
  - INQUIRE TEMPSTORAGE, SET TEMPSTORAGE
- **Existing SPI commands made threadsafe:**
  - INQUIRE CLASSCACHE
  - INQUIRE JVM
  - INQUIRE JVMPOOL
  - INQUIRE JVMPROFILE
  - PERFORM CLASSCACHE
  - PERFORM JVM POOL
  - SET CLASSCACHE
  - SET JVMPOOL




All new SPI commands are threadsafe and some existing SPI commands are also now threadsafe.

## Open transaction environment settings (1 of 2)

- **Separated out whether an application must run on an open TCB from what type of APIs it uses**
- **CONCURRENCY(REQUIRED)**
  - Application must be coded to threadsafe standards
  - States that the application MUST run on an open TCB
  - Application starts on an open TCB
  - If a switch to QR is made for a CICS command, a switch back to the open TCB is made when returning to the application
- **Existing API keyword defines what APIS are used**
  - This defines what type of TCB is used




In this release, you can now separate out whether an application can run on an open TCB from what type of APIs it uses. The CONCURRENCY attribute has a new value of REQUIRED. This value defines that the application is coded to threadsafe standards and that it must run on an open TCB. The application starts on an open TCB. If a switch to the QR TCB occurs for a CICS command, CICS switches back to the open TCB when returning to the application. The existing API attribute defines that APIs are used, which defines what type of open TCB is used.

## Open transaction environment settings (2 of 2)

- **CONCURRENCY(REQUIRED) and API(CICSAPI)**
  - The application will run on an open TCB from the start.
  - It only uses CICS supported APIs (including DB2, IMS® and WebSphere MQ)
  - CICS always use an L8 TCB in this instance irrespective of the execution key.
  - Great for applications that are going to resource managers like DB2 and WebSphere MQ as the same L8 is used
- **CONCURRENCY(REQUIRED) and API(OPENAPI)**
  - The application will run on an open TCB from the start.
  - It uses non CICS APIs
  - Application runs on an L8 or an L9 TCB depending on the execution key. This is the same as CICS TS 4.1
  - Only use OPENAPI when non CICS supported APIs are to be used



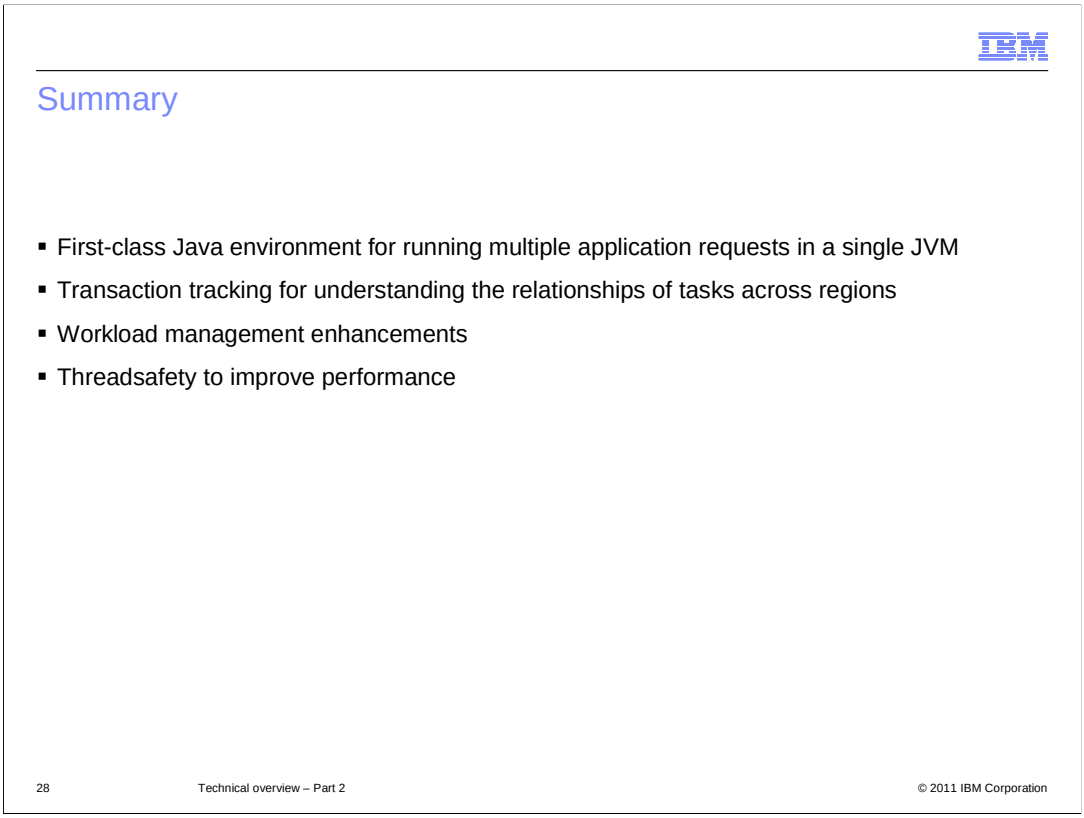

Here are two examples to show the separation. If you define CONCURRENCY as REQUIRED and the API attribute as CICSAPI, the application runs on an open TCB from the start. The application only uses CICS APIs. In this example, CICS always uses an L8 TCB because CICS APIs do not rely on the execution key of the TCB. This is the optimal setting for applications that are going to resource managers like DB2 and WebSphere MQ as the same L8 TCB is used. If you define CONCURRENCY as REQUIRED and the API attribute as OPENAPI, the application runs on an open TCB from the start. However, as the application is going to use non CICS APIs, it runs on an L8 or L9 TCB depending on the execution key. This is the same behavior as CICS TS 4.1.

## Threadsafe mirror

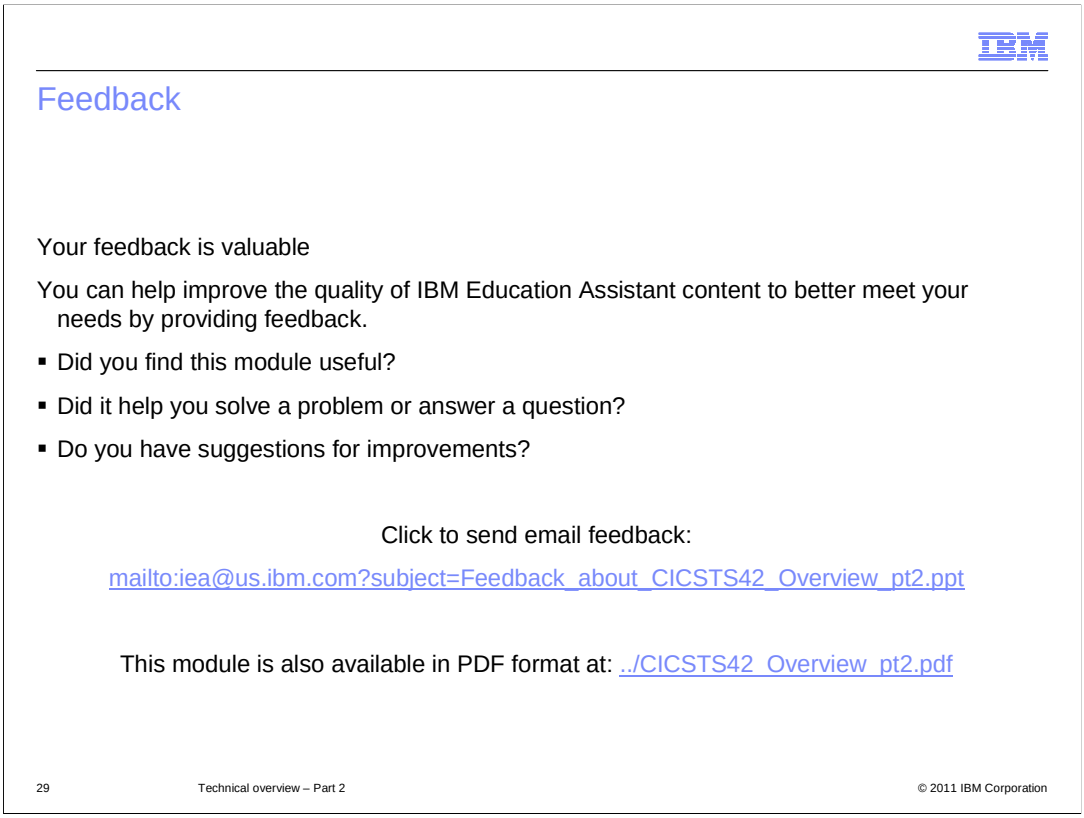
- **DFHMIRS is now threadsafe**
  - Supplied definition now specifies CONCURRENCY(THREADSAFE)
- **IPIC transformers are now threadsafe. Non IPIC code remains non threadsafe**
- **Only requests function shipped over IPIC will run on an Open TCB**
  - File Control
  - Temporary Storage
  - Distributed Program Link (DPL)
    - If the target program is defined as threadsafe and the mirror already on an open TCB
- **Review your DFHSIT specification for FCQRONLY**
  - Specify FCQRONLY=NO as there is no longer any need to turn off threadsafety in the FOR




The mirror program is now also threadsafe and the supplied definition has been updated. IPIC transformers are also threadsafe. For IPIC connections only, CICS runs DFHMIRS on an L8 open TCB whenever possible. Check your system initialization settings, as you do not have to turn off threadsafety in the file-owning region.




In summary, CICS provides a first-class Java environment for running multiple concurrent requests for an application in a single JVM. The Explorer SDK, available to download into any Eclipse IDE and in Rational Developer for System z, provides support for developing and deploying Java applications for any in-service release of CICS using the OSGi service platform standard. Transaction tracking provides a mechanism for understanding the relationship and chaining between transactions across many CICS regions. Workload management enhancements provide more flexibility when managing application workloads. Finally, the additional threadsafe commands improve the performance of applications by running in the open transaction environment and minimizes TCB switching.




You can help improve the quality of IBM Education Assistant content by providing feedback.

# Trademarks, disclaimer, and copyright information