CICS Transaction Server for z/OS

A Rexx/MVS interface to CICS using EXCI

Robert Harris,
CICS Technical Strategist,
IBM Hursley.

Issued: 31 August 2001
Revision Date: 31 August 2001
Previous Revision Date: None
Review Date: As Required

Take Note!

Before using this document be sure to read the general information under "Notices".

First Edition, August 2001.

Notices:

The following paragraph does not apply in any 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 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.

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates.

Any reference to an IBM licensed program or other IBM product in this publication is not intended to state or imply that only IBM's program or other product may be used. Any functionally equivalent program that does not infringe any of the intellectual property rights may be used instead of the IBM product.

Evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, is the user's responsibility.

IBM may have patents or pending patent applications covering subject matter 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 the IBM Director of Licensing, IBM Corporation, 500 Columbus Avenue, Thornwood, New York 10594, USA.

The information contained in this document has not be submitted to any formal IBM test and is distributed AS-IS. The use of the information or the implementation of any of these techniques is a customer responsibility and depends on the customer's ability to evaluate and integrate them into the customer's operational environment. While each item has been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

Trademarks:

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

3090	ACF/VTAM	AD/Cycle
AFP	AIX	AnyNet
Application System/400	APPN	AS/400
AT	BookManager	C Set++
C/370	C/MVS	CBIPO
CBPDO	CICS	CICS/400
CICS/6000	CICS/ESA	CICS/MVS
CICS OS/2	CICS TS	CICS/VM
CICS/VSE	CICSPlex	CICSPlex SM
COBOL/370	COBOL/2	Common User Access
CUA	DATABASE 2	DB2
DFSMS	DFSMS/MVS	DFSMSdfp
DFSMSdss	DFSMShsm	DFSORT
DXT	eNetwork	Enterprise Systems Architecture/370
Enterprise Systems Architecture/390	ES/3090	ESA/370
ESA/390	ES/9000	ESCON
GDDM	HiperBatch	Hiperspace
InfoWindow	IBM	IBMLink
IMS	IMS/ESA	Language Environment
MQ	MQSeries	MVS
MVS/DFP	MVS/ESA	MVS Parallel Sysplex
MVS/SP	MVS/XA	Multiprise
NetView	OpenEdition	OS/2
OS/390	OS/400	Processor Resource/Systems Manager
Parallel Sysplex	PR/SM	Presentation Manager
RACF	Resource Measurement Facility	RETAIN
RISC System/6000	RMF	RT
S/370	S/390	SAA
SQL/DS	SP	System/36
System/38	System/360	System/370
System/390	SystemView	Systems Application Architecture
VisualAge	VSE/ESA	VTAM
WebExplorer	z/OS	

UNIX is a registered Trademark in the United States and other countries licensed exclusively through X/Open Company Limited

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries.

INTEL is a registered trademark of Intel Corporation, in the United States, or other countries, or both.

Microsoft, Windows, and Windows NT are trademarks of Microsoft Corporation in the United States, or other countries, or both.

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

Summary of amendments

Date Of Change	Change made to document
31/08/2001	Creation in SupportPac CA1I

Reference Material and Bibliography:

This document uses a short reference to the following documentation:

Short reference	Book Title
CICS RDO Book	CICS Resource Definition Guide SC34-5722
CICS SDG	CICS System Definition Guide SC34-5725
CICS CG	CICS Customization Guide SC34-5706
CICS EXT	CICS External Interfaces Guide SC33-1944
CICS APR	CICS Application Programming Reference SC34-5702
CICS SPR	CICS System Programming Reference SC34-5726

Other SupportPacs:

The following SupportPacs contain information relevant to this SupportPac:

SupportPac	Relevance
CA1J	A SupportPac that uses this interface within the MVS BatchPipes environment

Preface:

This SupportPac consists of a utility that enables Rexx/MVS access to CICS function. It does this by running an EXCI Connection to a CICS Transaction Server for z/OS region.

This SupportPac contains information on:

- Configuring CICS for EXCI usage
- Rexx function to run an EXCI connection to CICS

It can be used to:

- Incorporate CICS-sourced information into ISPF processing
- Include CICS-sourced information in (TSO) Batch Processing
- Provide linkages between CICS and Netview for automation purposes

You need to understand RDO configuration for CICS and have a general familiarity with Rexx MVS programming to get the best out of this SupportPac.

The information and code in this SupportPac is **only** applicable to CICS Transaction Server for z/OS. It is not applicable to earlier CICS releases.

Table of Contents

Chapter 1: Introduction

1.1: The Concept

This SupportPac provides a method for a Rexx/MVS Exec to access CICS Transaction Server for z/OS facilities via the EXCI Interface.

This permits a Rexx/MVS Exec to issue a CICS Distributed Program link to a CICS region in the same MVS. A named program is executed within the CICS region with data sent from Rexx within a Commarea. The results of the execution are returned from CICS in the Commarea and made available to the caller as Rexx Variables.

Chapter 2: Installation

2.1: CICS prerequisites

The code in this SupportPac uses various standard CICS EXCI facilities. You must have configured the CICS region for this type of access. The things you have to do for this are contained in the *'External CICS Interface'* part of the *CICS EXT* book.

In particular, you must:

- Code the required DFHXCOPT EXCI Interface table
- Code up an optional DFHXCURM User-replaceable module
- Supply RDO definitions for EXCI Connections

2.1.1: Coding the DFHXCOPT Table

The DFHXCOPT table defines the connectional characteristics of the EXCI linkage to CICS. Although it controls tracing and other debugging facilities, the main usage to name the SVC being used for communication with CICS. The SVC number to use is that given for **`CICSSVC`** in the SIT.

See *'External CICS interface options table, DFHXCOPT'* in the *CICS EXT* for full information including how to compile the table.

The source for your DFHXCOPT should look something like:

```
DFHXCO TYPE=CSECT,                                                    |
       CICSSVC=245,                                                   |
       CONFDATA=SHOW,                                                 |
       DURETRY=30,                                                    |
       GTF=ON,                                                        |
       MSGCASE=MIXED,                                                 |
       SURROGCHK=NO,                                                  |
       TIMEOUT=60000,                                                 |
       TRACE=2,                                                       |
       TRACESZE=16,                                                   |
       TRAP=OFF
END DFHXCOPT
```

where SVC 245 is being used for communication.

This table must be assembled and linked into a library which is in the `//STEPLIB` concatination of the Rexx/MVS job.

2.1.2: Coding the DFHXCURM Module

The DFHXCURM is a CICS user-replaceable module that can be used to set defaults used for EXCI Communication. You do not have to code a DFHXCURM, but if you do so, it must be placed in the `//STEPLIB` concatination for the Rexx/MVS job.

Full details are in *'The EXCI user-replaceable module'* section of the *CICS EXT*.

The main use of DFHXCURM is to permit defaults to be specified for communication parameters or to police those that are supplied by the user.

However, DFHXCURM is not needed for normal usage of this SupportPac, and it is recommended that you do not supply one unless necessary.

2.1.3: Defining the EXCI RDO Connections

A RDO `CONNECTION` definition and its associated `SESSIONS` must be active in the region for Rexx usage of EXCI. You have to code these RDO items in a group and ensure that this Group in is a suitable `LIST` (or is manually installed).

The CONNECTION definition names the EXCI pathway between the Rexx/MVS Exec and the CICS region, whilst the SESSIONS define the number of active pathways. See the *CICS RDO* Book for full information on specifying CONNECTION and SESSIONS, and the *'Defining connections to CICS'* section of the *CICS EXT*.

This SupportPac uses a `GENERIC` EXCI connection to the CICS region. Consequently, the CONNECTION definition will look something like:

```
 CONnection     : REXX
 Group          : RAHEXCI
 DEscription    : LINKAGE FROM REXX
CONNECTION IDENTIFIERS
 Netname        :
 INDsys         :
REMOTE ATTRIBUTES
 REMOTESYSTem   :
 REMOTEName     :
 REMOTESYSNet   :
CONNECTION PROPERTIES
 ACcessmethod   : IRc               Vtam | IRc | INdirect | Xm
 PRotocol       : Exci              Appc | Lu61 | Exci
 Conntype       : Generic           Generic | Specific
 SInglesess     : No                No | Yes
 DAtastream     : User              User | 3270 | SCs | STrfield | Lms
 RECordformat   : U                 U | Vb
 Queuelimit     : No                No | 0-9999
 Maxqtime       : No                No | 0-9999
OPERATIONAL PROPERTIES
 AUtoconnect    : No                No | Yes | All
 INService      : Yes               Yes | No
SECURITY
 SEcurityname   :
 ATtachsec      : Local             Local | Identify | Verify | Persistent
                                    | Mixidpe
 BINDPassword   :                   PASSWORD NOT SPECIFIED
 BINDSecurity   : No                No | Yes
 Usedfltuser    : No                No | Yes
RECOVERY
 PSrecovery     :                   Sysdefault | None
 Xlnaction      : Keep              Keep | Force
```

The associated SESSIONS definition will look like:

```
 Sessions       : RAHEXCI
 Group          : RAHEXCI
 DEscription    : LINK TO REXX
SESSION IDENTIFIERS
 Connection     : REXX
 SESSName       :
 NETnameq       :
 MOdename       :
SESSION PROPERTIES
 Protocol       : Exci               Appc | Lu61 | Exci
 MAximum        : 000 , 000          0-999
 RECEIVEPfx     : RX
 RECEIVECount   : 005                1-999
 SENDPfx        :
 SENDCount      :                    1-999
 SENDSize       : 04096              1-30720
 RECEIVESize    : 04096              1-30720
 SESSPriority   : 000                0-255
 Transaction    :
OPERATOR DEFAULTS
 OPERId         :
 OPERPriority   : 000                0-255
 OPERRsl        : 0
 OPERSecurity   : 1
PRESET SECURITY
 USERId         :
OPERATIONAL PROPERTIES
 Autoconnect    : All                No | Yes | All
 INservice      : Yes                No | Yes
 Buildchain     : Yes                Yes | No
 USERArealen    : 000                0-255
 IOarealen      : 04096 , 04096      0-32767
 RELreq         : No                 No | Yes
 DIscreq        : No                 No | Yes
 NEPclass       : 000                0-255
RECOVERY
 RECOVOption    : Sysdefault         Sysdefault | Clearconv | Releasesess
                                     | Uncondrel | None
 RECOVNotify    : None               None | Message | Transaction
```

Observe that this definition only has Receive-type terminals and 4k is used for buffering of the flows. 5 concurrent connections into CICS are allowed.

2.2: Installing the SupportPac

2.2.1: Unpacking the ZIP file

Unzip `the CA1I.ZIP` file. It contains:

- `CA1IMODR.BIN` a module that provides the EXCI interface for a Rexx/MVS Exec for this SupportPac in an unloaded MVS PDS load module format

The SupportPac contains a single Rexx/MVS module called Exec called `RXDPL`.

2.2.2: Copying the PDS to MVS

You should copy the unloaded MVS PDS **CA1IMODR.BIN** to MVS.

You can do this operation via file transfer under TSO. Follow these instructions exactly or else the transfer will not work (It is assumed that IBM Personal Communications is being used as the 3270 emulator).

- Define (if not already present) a transfer type with the ASCII, CRLF and APPEND checkboxes unselected, the LRECL set to 80 with the transfer using FIXED length records; I call this the LOADLIB transfer type
- Transfer the `CA1IMODR.BIN` file to MVS with a file name of CA1IMODC using the LOADLIB transfer type. This will create a flat file called `CA1IMODC` in MVS
- In TSO, issue a RECEIVE INDSN(CA1IMODC); when prompted for a dataset name reply DSN(CA1IMODR). A PDS called CA1IMODR should be created in MVS containing the RXDPL module.

Alternatively, you can use FTP to transfer the PDS.

- Use the `FTP` command to start a session to the MVS region
  - Issue a `BINARY` command to prevent corruption of the data
  - Issue a `QUOTE SITE RECFM=FB` to transfer in the correct layout
  - Issue a `QUOTE SITE LRECL=80` to transfer in the correct layout
  - Send the file via `PUT CA1IMODR.BIN CA1IMODC`
- In TSO, issue a RECEIVE INDSN(CA1IMODC); when prompted for a dataset name reply DSN(CA1IMODR). A PDS called CA1IMODR should be created in MVS containing the RXDPL module.

Once this `CA1IMODR` PDS has been created, copy the RXDPL member to a Loadlib in the `//STEPLIB` concatination for your Rexx/MVS Exec.

2.2.3: Installing within TSO

You must include in the `//STEPLIB` concatination the CICS-supplied `SDFHEXCI` library which contains the EXCI interface routines and the library that contains the `DFHXCOPT` table.

There is nothing special that has to be done within the TSO (either ISPF or Batch) environment to invoke the `RXDPL` module that provides the function for this interface (there is no equivalent in REXX/MVS to the workstation versions that require a `RxFuncAdd` call for access to external modules).

As long as `RXDPL` is placed in a Loadlib which is in the `//STEPLIB` concatination all should be well.

The JCL for a batch job should look like:

```
//EXCIRUN  EXEC PGM=IKJEFT01
//SYSPROC  DD  DSN=<library containing your exec>,DISP=SHR
//STEPLIB  DD  DSN=<library containing RXDPL>,DISP=SHR,DCB=BLKSIZE=32760
//         DD  DSN=<library containing DFHXCOPT>,DISP=SHR
//         DD  DSN=CICS.SDFHEXCI,DISP=SHR
//SYSPRINT DD  SYSOUT=*
//SYSUDUMP DD  SYSOUT=*
//SYSTSPRT DD  SYSOUT=*,DCB=(RECFM=F,LRECL=132,BLKSIZE=132)
//SYSTSIN  DD  *
<<your exec invocation>>
/*
```

Chapter 3: Operational Characteristics of RXDPL

3.1: Calling RXDPL

RXDPL is called in the manner of an external subroutine:

```
rcc = RXDPL('command' , p1, p2, p3 )
```

The first parameter is the name of a function for the interface.

RXDPL always returns a string representing the Return Code for the operation. The first few words in this Return Code String are fixed:

1	The overall interface Return Code (0 is OK)
2	A functional Return Code (this will be 0 if word 1 is generated by RXDPL itself)
3	A functional Reason Code (this will be 0 if word 1 is generated by RXDPL itself)
4	The function being executed: RXDPLINIT - Initialisation RXDPLLINK - DPL function not involving a flow EXCI - DPL function involving a flow RXDPLTERM - Termination RXDPL - Unknown request

The overall Return Codes have Rexx literals available for use. These are listed in Section 8 "Literals placed in the Rexx workspace" on page 25.

3.2: Rexx Stem Variables

This SupportPac uses Rexx Stem variables for passing information to and from CICS. A Stem variable is in the format of `name.component` : the `name.` part can be whatever desired, but the `component` part (after the dot) is defined by the interface.

Normal REXX variable substitutions apply to the stem variable, and the RXDPL module folds all names into Uppercase before usage. Consequently, the Stem variable will be specified to the interface as a literal `'stem.' .`

You can cause conflicts if you use a component name used by RXDPL as a normal variable. This is because the normal Rexx substitution rules apply to component names, so the substituted value will be passed to RXDPL instead of what is expected.

```
a.1 = 5 ; a.2 = 3
b = 2
say a.b

/* produces 3 due to the component substitution */
```

3.3: Return Codes

The numeric Return Code for the operation can be extracted from the Return Code string by doing a

```
lastrc = strip(word(rcc,1),'B')
```

operation. These return codes can be generated by RXDPL (negatives), or the EXCI Interface (generally >200) or CICS itself.

A Stem variable called `RXDPL.RCMAP.n` converts these numbers into a readable text and another Stem variable RXDPL.x provides a mapping of EXCI names into their numeric equivalents (these are listed in Section 8 "Literals placed in the Rexx workspace" on page 25).

```
rcc = RXDPL('function', p1, p2)
lastrc = strip(word(rcc,1),'B')
lastfailure = RXDPL.RCMAP.lastrc
```

Return Codes classified by EXCI as Warnings (those <100) are treated by RXDPL as OKs, and not returned.

3.4: Trace

Trace is enabled by setting the `RXDPLTRACE` variable. Trace output goes to the standard destination (for the current environment) that the `say` operation uses (which is usually `//SYSTSPRT`). If you are using RXDPL within a procedure, you need to `expose RXDPLTRACE` accordingly. Trace is initially off.

The available settings follow functions within RXDPL:

`COMMON`	`Things executed on each usage of RXDPL`
`INIT`	`Execution of this initialise function for the interface`
`TERM`	`Execution of this termination function for the interface`
`LINK`	`Execution of actual DPL flows by the interface`
`SV`	`Setting Rexx Variables`
`GV`	`Reading Rexx variables`
`*`	`Trace everything`

In general, if you are using Trace, the `*` setting should be used.

```
RXDPLTRACE = '*'
```

3.5: Use of EXCI facilities

RXDPL uses the six native EXCI interfaces as defined in the *CICS EXT* book under the *The EXCI Call Interface* section.

A new pipe is obtained for each usage of RXDPL (so a single pipe is **not** used for multiple usages within an Exec).

If one of the `Initialize_User`, `Allocate_Pipe` or `Open_Pipe` calls fail, then subsequent ones will not be called and the `DPL_Request` flow will not be made. In all cases, an attempt will be made to cleanly end EXCI communication by issuing the `Close_Pipe` and `Deallocate_Pipe` calls.

Chapter 4: Running RXDPL

The RXDPL interface is a module that acts as an external Rexx/MVS subroutine. There is only one call (`RXDPL`) with the function required being specified as the first parameter.

There are three calls for this interface:

- Initialisation
- Link
- Termination

4.1: Initialisation

This function initialises the RXDPL interface and puts all the `RXDPL.` mappings into the Rexx variable pool. These mappings are listed in Section 8 "Literals placed in the Rexx workspace" on page 25.

This function should only be executed once within an Exec.

```
rcc = RXDPL('INIT')
```

4.1.1: Return Code String

The first five words of the Return Code string for the Initialise function will usually be

```
0 0 0 RXDPLINIT OK
```

4.2: Link

This call does the EXCI linkage to CICS. Each invocation allocates an EXCI Pipe, issues the DPL call and terminates the Pipe. Full details of the APIs used are in the *CICS EXT* book under the *The EXCI Call Interface* section. The interface uses the six individual calls and not the generic `EXEC CICS CALL` function.

This call takes three parameters:

```
rcc = RXDPL('LINK', 'ControlStem.' ,
                    'InputCommareaStem.',
                    'OutputCommareaStem.')
```

The first parameter `ControlStem.` is the **name** of a Stem Variable (including the dot) whose components define the characteristics and destination of the EXCI flow to CICS (things like the Applid and Program).

The second parameter `InputCommareaStem.` is the **name** of a Stem Variable (including the dot) which contains the data to be sent to CICS in a Commarea. `InputCommareaStem.0` contains the length of the data which is in `InputCommareaStem.1`.

The third parameter `OutputCommareaStem.` is the **name** of a Stem Variable (including the dot) into which the Commarea sent from CICS as a result of the EXCI flow is to be placed. `OutputCommareaStem.0` will contain the length of the returned Commarea which is in `OutputCommareaStem.1` . It is updated with additional components showing the results of the EXCI flow.

As all these parameters are names of Stem variables, they should be specified as literals (in quotes) to avoid unwarranted substitutions.

If, due to an error, the physical commarea used for the EXCI Communication to CICS is **not** updated by the EXCI layers, the Input Commarea will be returned.

4.2.1: Specifying and using the Control Stem

The Control Stem variable defines the flow to CICS. It is not updated by RXDPL and so can be reused for another interface usage.

The following components are used to define the EXCI flow:

APPLID	The 8 byte VTAM Applid of the CICS region to process the EXCI flow. If this component is omitted (or specified as blanks) this parameter will not be used in the EXCI Interface call (Allocate_Pipe) and so the DFHXCURM module is assumed to supply the required destination
PROG	The 8 byte PROGRAM that is to be executed within the CICS region to service the DPL request. This component must be specified. If it is invalid, CICS will return an exception upon execution of the EXCI code in the CICS region.
TRAN	The 4 byte CICS Transaction ID representing the Mirror Transaction that is to run in CICS to process the EXCI flow. If this component is omitted (or specified as blanks) this parameter will not be used in the EXCI Interface call (DPL_Request). This will cause CICS to use CSMI as the mirror transaction. However, the DFHXCURM module can fixup this transaction if so desired.
USERID	The 8 byte Userid that will run the DPL function within CICS. This component must be specified. The usual way is to get the setting from the REXX USERID() function. If this is invalid, or results in a Security Violation, CICS will reject the EXCI call.

The Control Stem will look like:

```
drop cs.
cs.prog   = 'RAHTEST1'
cs.applid = 'IYCKRAH6'
cs.userid = userid()
cs.tran   = 'CSMI'

.... RXDPL('LINK', 'cs.' , ....)
```

4.2.2: Commareas

The Input and Output Commarea Stems follow the usual convention for variable length Rexx Items: `.0` contains the length of data in `.1` .

The `OutputCommareaStem.` can have no components on RXPL calling. In this case, the length of the physical commarea used by RXDPL for EXCI traffic will be the length contained in `InputCommareaStem.0` . If `OutputCommareaStem.0` contains a value, then the length of the physical commarea used by RXDPL will be the maximum of the Input and Output Commarea lengths. In both cases the actual length of the data sent via EXCI to CICS will be the length of the Input Commarea as specified in `InputCommareaLength.0` (the `data_len` setting of the `DPL_Request` call).

The length of an Input or Output Commarea must be between 1 and 32500 bytes

```
32500 >= Input or Output Commarea Length in <name>Stem.0 >= 1
```

After the EXCI call, the contents of the Return Commarea (all of the physical commarea used by RXDPL) will be placed in `OutputCommareaStem.1` with the length of this data in `OutputCommareaStem.0` .

4.2.3: Input Commarea Stem

The Input Commarea Stem is not updated by `RXDPL`. Consequently, it can be reused for another DPL flow.

The InputCommareaStem must have the length of data held in .1 contained in .0 when RXDPL is called.

No checks are made to see that the actual length of the information in `InputCommareaStem.1` corresponds to the length specified in `InputCommareaStem.0`.

4.2.4: Output Commarea Stem

New components are added after the EXCI flow has occurred to show the status of the request. They will **not** be present unless an EXCI Flow made it to CICS. Errors such as bad parameter specifications or CICS not being contactable will **not** cause these components to be generated. The additional components are:

DIDFLOW	Shows whether or not an EXCI flow was actually sent to CICS and whether it succeeded or not. The settings are Y — A Flow made it to CICS and this was successfully processed F — A Flow made it to CICS but this resulted in an Error (or an Abend) not present or set to DIDFLOW — A Flow did not make it to CICS either due to an error or CICS not being contactable (bad APPLID)
0	The length of the returned Commarea This will only be updated if an EXCI flow was actually sent to CICS and CICS returned a Commarea. Parameter violations or the specification of a non-contactable CICS will not update this component.
1	The actual returned Commarea This will only be present if an EXCI flow was actually sent to CICS and CICS returned a commarea. Parameter violations or the specification of a non-contactable CICS will not generate this component
AC	The EXCI Reason Code This is taken from Field 2 of the dpl_retarea and shows the Exec RESP2 for the DPL program execution
RESP	The EXEC Return Code This is taken from Field 1 of the dpl_retarea and shows the value of EIBRESP for the DPL operation of the program
RESP2	The EXEC Reason Code This is taken from Field 2 of the dpl_retarea and shows the value of EIBRESP2 for the DPL operation of the program
ABEND	The EXEC Abend Code (blanks if no abend occurred) This is taken from Field 3 of the dpl_retarea and shows the abend code that the DPL operation of the program possibly engendered.
MSG	Any message that CICS returned as a result of the DPL flow. This is taken from the 5 word Return Area for the dpl_request call. It will be set to a blank if CICS did not send any message. If CICS sent a message, it will start in the usual fashion with the first word being the Message Number and the next two words being the date and time: the actual message starts at word 4.

When RXDPL is called, only `OutputCommareaStem.0` can be optionally supplied. No other components should be present.

The `OutputCommareaStem` contains the results of the EXCI flow - whether or not this reached CICS and whether or not the flow succeeded.

When RXDPL is called, the `OutputCommareaStem.1` (representing the returned data) should not be present. However, if this is supplied, it will **not** be deleted if an error occurred. This is so that a Rexx Exec can test the `OutputCommareaStem.1` to see if it has changed and so deduce whether or not the EXCI process has succeeded.

Similarly, the `OutputCommareaStem.0` component may not be updated to the actual length of the physical Commarea being used for EXCI traffic if an error has occurred.

The `OutputCommareaStem.` should be `drop`ed before a multiple usage of RXDPL in an Exec to avoid processing errors associated with the prior flow.

4.2.5: Unit of Work Considerations

RXDPL will never associate itself with the MVS (or any other) Recovery Manager. All flows to CICS are made with the `SYNCONRETURN` setting.

4.2.6: Connection and CICS Considerations

RXDPL uses a Generic EXCI Connection to CICS (see Section 2.1.3 "Defining the EXCI RDO Connections" on page 5).

The relevant CICS region must:

- Have the required RDO definitions active (see Section 2.1.3 "Defining the EXCI RDO Connections" on page 5)
- Be accepting VTAM traffic (`CEMT SET VTAM OPEN`)
- Be accepting MRO Traffic (`CEMT SET IRC OPEN`)
- Have the Mirror Transaction and DPLed Program active and available
- Be authorised to accept function for the relevant Userid

4.2.7: Return Codes from the linkage

In the case of the Linkage function, RXDPL returns a Return Code String depending upon how far function got before the error or unexpected circumstance arose.

The wholly successful function will return

```
0 0 0 RXDPLLINK OK
```

If an error arose before the EXCI function is attempted, the format will be

```
Error_Code 0 0 RXDPLLINK description_of_error
```

If the error arose due to EXCI processing, the format will be dependant upon where the error arose. However, it will start

```
Overall_Return_Code EIBRESP_code EIBRESP2_code EXCI
```

The fifth and subsequent words will be a description of the error. If any one of these subsequent words (5 and upwards) is a colon, then after the colon will be a message sent from CICS. This message will be in the usual format of MessageId, Date, Time and the actual text of the message.

```
... EXCI <more words> : MessageId Date Time the_message_itself
```

The easiest way to see whether or not an EXCI request was successfully executed within CICS is to check the `DIDFLOW` component of the Output Stem. If this is `'Y'` then the EXCI flow was successful.

```
drop dpcon.
dpcon.applid = 'IYCKRAH6' ; dpcon.prog   = 'RAHUT001'
dpcon.userid =  userid()

drop dpin. ; dpin.0 = 15 ; dpin.1 = 'RAH rules OK'

drop dpout.

rcc = RXDPL('LINK', 'dpcon.', 'dpin.', 'dpout.')
frc = strip(word(rcc,1),'B')

select
 when ( dpout.didflow = 'Y' ) do ; /* successful flow logic */ ; end
 when ( dpout.didflow = 'F' ) do ; /* CICS error in flow    */ ; end
 otherwise                    do ; /* did not make it       */ ; end
 end
```

4.3: Termination

This function terminates the RXDPL interface but leaves the `RXDPL.` mappings in the Rexx variable pool.

This function should only be executed once within an Exec.

```
rcc = RXDPL('TERM')
```

4.3.1: Return Code String

The first five words of the Return Code string for the Termination function will usually be

```
0 0 0 RXDPTERM OK
```

Chapter 5: Common Runtime Errors

This section shows returned variables for common runtime errors. Observe that CICS traps some of these and treats them as EXCI errors whilst others are deemed to be Application errors and so treated differently.

5.1: Unknown Mirror Transaction

This circumstance is treated as a correct EXCI flow which results in a system error.

rcc	0414 0000 0000 EXCI Flow DPL failure : DFHAC2001 31/08/2001 11:31:53 IYCKRAH6 Transaction 'CSMX' is not recognized. Check that the transaction name is correct.
didflow	F
AC	0414
RESP	0000
RESP2	0000
ABEND	' '
MESSAGE	DFHAC2001 31/08/2001 11:31:53 IYCKRAH6 Transaction 'CSMX' is not recognized. Check that the transaction name is correct.

5.2: Unknown Program

This circumstance occurs in the Mirror transaction so the failure is treated as an application error.

rcc	0 0 0 RXDPLLINK OK
didflow	F
AC	0000
RESP	0027
RESP2	0000
ABEND	' '
MESSAGE	' '

5.3: CICS not contactable

This is detected on the EXCI `Open_Pipe` call.

rcc	203 92 0 EXCI Open Pipe failure
didflow	<not defined>
AC	<not defined>
RESP	<not defined>
RESP2	<not defined>
ABEND	<not defined>
MESSAGE	<not defined>

5.4: Program Abended

This specific failure is trapped by EXCI and so results in an EXCI type of error.

rcc	0422 0000 0000 EXCI Flow DPL failure :
didflow	F
AC	0422
RESP	0000
RESP2	0000
ABEND	<the 4 byte Abend Code>
MESSAGE	' '

5.5: Bad Userid

This circumstance is treated as a correct EXCI flow which results in a system error.

rcc	0414 0000 0000 EXCI Flow DPL failure : DFHAC2047 31/08/2001 12:21:47 IYCKRAH6 While performing an attach for node DFHGEN a security violation was detected.
didflow	F
AC	0414
RESP	0000
RESP2	0000
ABEND	' '
MESSAGE	DFHAC2047 31/08/2001 12:21:47 IYCKRAH6 While performing an attach for node DFHGEN a security violation was detected.

Chapter 6: Example call

This example shows the information that has to be passed in the RXDPL call

Applid	The CICS region to contact is supplied in the ControlStem `dpcon.applid`
Program	The program to run is supplied in the ControlStem `dpcon.prog`
Userid	The identity to be used in running the program is supplied in the ControlStem `dpcon.userid`
Mirror Transaction	Defaults to CSMI
Input commarea	Is supplied in the `dpin.` stem variable `dpin.0` contains the length of the area which is in `dpin.1`
Output commarea	Is placed into the `dpout.` stem variable After the call, `dpout.0` will contain the length of the return commarea data which is in `dpout.1`
Physical commarea	As `dpout.0` is not specified, the physical commarea length used for EXCI traffic is that in `dpin.0`

.

```
drop dpcon.
dpcon.applid = 'IYCKRAH6'
dpcon.prog   = 'RAHUT001'
dpcon.userid =  userid()

drop dpin.
dpin.0 = 15
dpin.1 = 'RAH rules OK'

drop dpout.

rcc = RXDPL('LINK', 'dpcon.', 'dpin.', 'dpout.')
frc = strip(word(rcc,1),'B')

say 'EXCI linkage Return Code /'frc'/'
say 'Commarea returned L('dpout.0') /'dpout.1'/'
```

7.1: Errors generated by RXDPL

-1	0 0 RXDPLLINK	Bad number of Parms	4 parms must be specified for the EXCI linkage call
-2	0 0 RXDPLLINK	Control Stem Variable not supplied	The 2nd parameter must be supplied
-3	0 0 RXDPLLINK	Input Commarea variable not supplied	The 3rd parameter must be specified
-4	0 0 RXDPLLINK	Output Commarea Variable not supplied	The 4th parameter must be supplied
-5	0 0 RXDPLLINK	PROG component not supplied	The Control Stem. variable must have a PROG component supplied which must not start with X'00' or X'40’
-6	0 0 RXDPLLINK	USERID component not supplied	The Control Stem. variable must have an USERID component supplied which must not start with X'00' or X'40’
-7	0 0 RXDPLLINK	Input Commarea not supplied	The length of the Input Commarea (in `.0`) must not be 0
-8	0 0 RXDPLLINK	Input Commarea data not supplied	A `.0` component must be specified for the Input Commarea Stem
-9	0 0 RXDPLLINK	Input Commarea too big	The maximum Commarea size is 32500
-10	0 0 RXDPLLINK	Commarea zero length	An attempt is being made to use a zero length physical commarea
-11	0 0 RXDPLLINK	Commarea area Getmain failure	Internal GETMAIN error in obtaining storage for the physical Commarea to be used for the EXCI flow
-98	0 0 RXDPL	Unknown Request	The first parameter is the function required and must be `'INIT'`, `'TERM'` or `'LINK'`
-99	0 0 RXDPL	Incorrect Number of parms supplied	At least some parameters must be supplied to RXDPL

7.2: Errors generated by EXCI

8	ERROR
16	INVREQ
22	LENGERR
27	PGMIDERR
53	SYSIDERR
70	NOTAUTH
81	TERMERR
82	ROLLEDBACK
88	LINKERR
201	NO_CICS_IRC_STARTED
202	NO_PIPE
203	NO_CICS
204	WRONG_MVS_FOR_RRMS
205	RRMS_NOT_AVAILABLE
401	INVALID_CALL_TYPE
402	INVALID_VERSION_NO
403	INVALID_APPL_NAME
404	INVALID_USER_TOKEN
405	PIPE_NOT_CLOSED
406	PIPE_NOT_OPEN
407	INVALID_USERID
408	INVALID_UOWID
409	INVALID_TRANSID
410	DFHMEBM_LOAD_FAILED
411	DFHMET4E_LOAD_FAILED
412	DFHXCURM_LOAD_FAILED
413	DFHXCTRA_LOAD_FAILED
414	IRP_ABORT_RECEIVED
415	INVALID_CONNECTION_DEFN
416	INVALID_CICS_RELEASE
417	PIPE_MUST_CLOSE
418	INVALID_PIPE_TOKEN
419	CICS_AFCB_PRESENT
420	DFHXCOPT_LOAD_FAILED
421	RUNNING_UNDER_AN_IRB
422	SERVER_ABENDED
423	SURROGATE_CHECK_FAILED
424	RRMS_NOT_SUPPORTED
425	UOWID_NOT_ALLOWED
426	INVALID_TRANSID2
427	INVALID_CCSID
428	INVALID_ENDIAN
601	WS_GETMAIN_ERROR
602	XCGLOBAL_GETMAIN_ERROR
603	XCUSER_GETMAIN_ERROR
604	XCPIPE_GETMAIN_ERROR
605	VERIFY_BLOCK_GM_ERROR
606	SSI_VERIFY_FAILED
607	CICS_SVC_CALL_FAILURE
608	IRC_LOGON_FAILURE
609	IRC_CONNECT_FAILURE
610	IRC_DISCONNECT_FAILURE
611	IRC_LOGOFF_FAILURE
612	TRANSFORM_1_ERROR
613	TRANSFORM_4_ERROR
614	IRP_NULL_DATA_RECEIVED
615	IRP_NEGATIVE_RESPONSE
616	IRP_SWITCH_PULL_FAILURE
617	IRP_IOAREA_GM_FAILURE
619	IRP_BAD_IOAREA
620	IRP_PROTOCOL_ERROR
621	PIPE_RECOVERY_FAILURE
622	ESTAE_SETUP_FAILURE
623	ESTAE_INVOKED
624	SERVER_TIMEDOUT
625	STIMER_SETUP_FAILURE
626	STIMER_CANCEL_FAILURE
627	INCORRECT_SVC_LEVEL
628	IRP_LEVEL_CHECK_FAILURE
629	SERVER_PROTOCOL_ERROR
630	RRMS_ERROR
631	RRMS_SEVERE_ERROR
632	XCGUR_GETMAIN_ERROR
903	ESTAE_SETUP_ERROR
904	ESTAE_INVOKED

Chapter 8: Literals placed in the Rexx workspace

RXDPL.CICS_AFCB_PRESENT	419
RXDPL.CICS_SVC_CALL_FAILURE	607
RXDPL.DFHMEBM_LOAD_FAILED	410
RXDPL.DFHMET4E_LOAD_FAILED	411
RXDPL.DFHXCOPT_LOAD_FAILED	420
RXDPL.DFHXCTRA_LOAD_FAILED	413
RXDPL.DFHXCURM_LOAD_FAILED	412
RXDPL.ERROR	8
RXDPL.ESTAE_INVOKED	623
RXDPL.ESTAE_INVOKED	904
RXDPL.ESTAE_SETUP_ERROR	903
RXDPL.ESTAE_SETUP_FAILURE	622
RXDPL.INCORRECT_SVC_LEVEL	627
RXDPL.INVALID_APPL_NAME	403
RXDPL.INVALID_CALL_TYPE	401
RXDPL.INVALID_CCSID	427
RXDPL.INVALID_CICS_RELEASE	416
RXDPL.INVALID_CONNECTION_DEFN	415
RXDPL.INVALID_ENDIAN	428
RXDPL.INVALID_PIPE_TOKEN	418
RXDPL.INVALID_TRANSID	409
RXDPL.INVALID_TRANSID2	426
RXDPL.INVALID_UOWID	408
RXDPL.INVALID_USER_TOKEN	404
RXDPL.INVALID_USERID	407
RXDPL.INVALID_VERSION_NO	402
RXDPL.INVREQ	16
RXDPL.IRC_CONNECT_FAILURE	609
RXDPL.IRC_DISCONNECT_FAILURE	610
RXDPL.IRC_LOGOFF_FAILURE	611
RXDPL.IRC_LOGON_FAILURE	608
RXDPL.IRP_ABORT_RECEIVED	414
RXDPL.IRP_BAD_IOAREA	619
RXDPL.IRP_IOAREA_GM_FAILURE	617
RXDPL.IRP_LEVEL_CHECK_FAILURE	628
RXDPL.IRP_NEGATIVE_RESPONSE	615
RXDPL.IRP_NULL_DATA_RECEIVED	614
RXDPL.IRP_PROTOCOL_ERROR	620
RXDPL.IRP_SWITCH_PULL_FAILURE	616
RXDPL.LENGERR	22
RXDPL.LINKERR	88
RXDPL.NO_CICS	203
RXDPL.NO_CICS_IRC_STARTED	201
RXDPL.NO_PIPE	202
RXDPL.NONE	0
RXDPL.NORMAL	0
RXDPL.NOTAUTH	70
RXDPL.OK	0
RXDPL.PGMIDERR	27
RXDPL.PIPE_MUST_CLOSE	417
RXDPL.PIPE_NOT_CLOSED	405
RXDPL.PIPE_NOT_OPEN	406
RXDPL.PIPE_RECOVERY_FAILURE	621
RXDPL.ROLLEDBACK	82
RXDPL.RRMS_ERROR	630
RXDPL.RRMS_NOT_AVAILABLE	205
RXDPL.RRMS_NOT_SUPPORTED	424
RXDPL.RRMS_SEVERE_ERROR	631
RXDPL.RUNNING_UNDER_AN_IRB	421
RXDPL.SERVER_ABENDED	422
RXDPL.SERVER_PROTOCOL_ERROR	629
RXDPL.SERVER_TIMEDOUT	624
RXDPL.SSI_VERIFY_FAILED	606
RXDPL.STIMER_CANCEL_FAILURE	626
RXDPL.STIMER_SETUP_FAILURE	625
RXDPL.SURROGATE_CHECK_FAILED	423
RXDPL.SYSIDERR	53
RXDPL.TERMERR	81

```
RXDPL.TRANSFORM_1_ERROR            612
RXDPL.TRANSFORM_4_ERROR            613
RXDPL.UOWID_NOT_ALLOWED            425
RXDPL.VERIFY_BLOCK_GM_ERROR        605
RXDPL.WARNING                      4
RXDPL.WRONG_MVS_FOR_RRMS           204
RXDPL.WS_GETMAIN_ERROR             601
RXDPL.XCGLOBAL_GETMAIN_ERROR       602
RXDPL.XCGUR_GETMAIN_ERROR          632
RXDPL.XCPIPE_GETMAIN_ERROR         604
RXDPL.XCUSER_GETMAIN_ERROR         603
```

```
RXDPL.RCMAP.0      RXDPL.NORMAL
RXDPL.RCMAP.4      RXDPL.WARNING
RXDPL.RCMAP.8      RXDPL.ERROR
RXDPL.RCMAP.16     RXDPL.INVREQ
RXDPL.RCMAP.22     RXDPL.LENGERR
RXDPL.RCMAP.27     RXDPL.PGMIDERR
RXDPL.RCMAP.53     RXDPL.SYSIDERR
RXDPL.RCMAP.70     RXDPL.NOTAUTH
RXDPL.RCMAP.81     RXDPL.TERMERR
RXDPL.RCMAP.82     RXDPL.ROLLEDBACK
RXDPL.RCMAP.88     RXDPL.LINKERR
RXDPL.RCMAP.201    RXDPL.NO_CICS_IRC_STARTED
RXDPL.RCMAP.202    RXDPL.NO_PIPE
RXDPL.RCMAP.203    RXDPL.NO_CICS
RXDPL.RCMAP.204    RXDPL.WRONG_MVS_FOR_RRMS
RXDPL.RCMAP.205    RXDPL.RRMS_NOT_AVAILABLE
RXDPL.RCMAP.401    RXDPL.INVALID_CALL_TYPE
RXDPL.RCMAP.402    RXDPL.INVALID_VERSION_NO
RXDPL.RCMAP.403    RXDPL.INVALID_APPL_NAME
RXDPL.RCMAP.404    RXDPL.INVALID_USER_TOKEN
RXDPL.RCMAP.405    RXDPL.PIPE_NOT_CLOSED
RXDPL.RCMAP.406    RXDPL.PIPE_NOT_OPEN
RXDPL.RCMAP.407    RXDPL.INVALID_USERID
RXDPL.RCMAP.408    RXDPL.INVALID_UOWID
RXDPL.RCMAP.409    RXDPL.INVALID_TRANSID
RXDPL.RCMAP.410    RXDPL.DFHMEBM_LOAD_FAILED
RXDPL.RCMAP.411    RXDPL.DFHMET4E_LOAD_FAILED
```

```
RXDPL.RCMAP.412    RXDPL.DFHXCURM_LOAD_FAILED
RXDPL.RCMAP.413    RXDPL.DFHXCTRA_LOAD_FAILED
RXDPL.RCMAP.414    RXDPL.IRP_ABORT_RECEIVED
RXDPL.RCMAP.415    RXDPL.INVALID_CONNECTION_DEFN
RXDPL.RCMAP.416    RXDPL.INVALID_CICS_RELEASE
RXDPL.RCMAP.417    RXDPL.PIPE_MUST_CLOSE
RXDPL.RCMAP.418    RXDPL.INVALID_PIPE_TOKEN
RXDPL.RCMAP.419    RXDPL.CICS_AFCB_PRESENT
RXDPL.RCMAP.420    RXDPL.DFHXCOPT_LOAD_FAILED
RXDPL.RCMAP.421    RXDPL.RUNNING_UNDER_AN_IRB
RXDPL.RCMAP.422    RXDPL.SERVER_ABENDED
RXDPL.RCMAP.423    RXDPL.SURROGATE_CHECK_FAILED
RXDPL.RCMAP.424    RXDPL.RRMS_NOT_SUPPORTED
RXDPL.RCMAP.425    RXDPL.UOWID_NOT_ALLOWED
RXDPL.RCMAP.426    RXDPL.INVALID_TRANSID2
RXDPL.RCMAP.427    RXDPL.INVALID_CCSID
RXDPL.RCMAP.428    RXDPL.INVALID_ENDIAN
RXDPL.RCMAP.601    RXDPL.WS_GETMAIN_ERROR
RXDPL.RCMAP.602    RXDPL.XCGLOBAL_GETMAIN_ERROR
RXDPL.RCMAP.603    RXDPL.XCUSER_GETMAIN_ERROR
RXDPL.RCMAP.604    RXDPL.XCPIPE_GETMAIN_ERROR
RXDPL.RCMAP.605    RXDPL.VERIFY_BLOCK_GM_ERROR
RXDPL.RCMAP.606    RXDPL.SSI_VERIFY_FAILED
RXDPL.RCMAP.607    RXDPL.CICS_SVC_CALL_FAILURE
RXDPL.RCMAP.608    RXDPL.IRC_LOGON_FAILURE
RXDPL.RCMAP.609    RXDPL.IRC_CONNECT_FAILURE
RXDPL.RCMAP.610    RXDPL.IRC_DISCONNECT_FAILURE
RXDPL.RCMAP.611    RXDPL.IRC_LOGOFF_FAILURE
RXDPL.RCMAP.612    RXDPL.TRANSFORM_1_ERROR
RXDPL.RCMAP.613    RXDPL.TRANSFORM_4_ERROR
RXDPL.RCMAP.614    RXDPL.IRP_NULL_DATA_RECEIVED
RXDPL.RCMAP.615    RXDPL.IRP_NEGATIVE_RESPONSE
RXDPL.RCMAP.616    RXDPL.IRP_SWITCH_PULL_FAILURE
RXDPL.RCMAP.617    RXDPL.IRP_IOAREA_GM_FAILURE
RXDPL.RCMAP.619    RXDPL.IRP_BAD_IOAREA
RXDPL.RCMAP.620    RXDPL.IRP_PROTOCOL_ERROR
```

```
RXDPL.RCMAP.621    RXDPL.PIPE_RECOVERY_FAILURE
RXDPL.RCMAP.622    RXDPL.ESTAE_SETUP_FAILURE
RXDPL.RCMAP.623    RXDPL.ESTAE_INVOKED
RXDPL.RCMAP.624    RXDPL.SERVER_TIMEDOUT
RXDPL.RCMAP.625    RXDPL.STIMER_SETUP_FAILURE
RXDPL.RCMAP.626    RXDPL.STIMER_CANCEL_FAILURE
RXDPL.RCMAP.627    RXDPL.INCORRECT_SVC_LEVEL
RXDPL.RCMAP.628    RXDPL.IRP_LEVEL_CHECK_FAILURE
RXDPL.RCMAP.629    RXDPL.SERVER_PROTOCOL_ERROR
RXDPL.RCMAP.630    RXDPL.RRMS_ERROR
RXDPL.RCMAP.631    RXDPL.RRMS_SEVERE_ERROR
RXDPL.RCMAP.632    RXDPL.XCGUR_GETMAIN_ERROR
RXDPL.RCMAP.903    RXDPL.ESTAE_SETUP_ERROR
RXDPL.RCMAP.904    RXDPL.ESTAE_INVOKED
```

>>>>>>>>>>>>>>>END of document<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<