CICS Transaction Server for z/OS



# Administering CICS

*Version 5 Release 4*

CICS Transaction Server for z/OS



# Administering CICS

*Version 5 Release 4*

**Note**
Before using this information and the product it supports, read the information in "Notices" on page 163.

This edition applies to the IBM CICS Transaction Server for z/OS Version 5 Release 4 (product number 5655-Y04) and to all subsequent releases and modifications until otherwise indicated in new editions.

**© Copyright IBM Corporation 1989, 2017.**
US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

# Contents


**About this PDF. . . . . v**

**Chapter 1. Controlling CICS operation. . 1**

Administering CICS from a console device . . . . 1
Entering commands from a console device . . . 2
Entering commands from TSO . . . . . 2
Using JCL to initiate CICS commands . . . . . 2
Console device messages . . . . . 3
Administering with CICS supplied transactions. . . 6
Starting or stopping a transaction . . . . . 6
Syntax notation and conventions used . . . . . 7
Terminal operation . . . . . 9
Using the system console. . . . . 14
Using TSO consoles . . . . . 16
Using the KILL option for purging transactions 17
Administering with the application debugging profile manager . . . . . 18
Using the application debugging profile manager Web interface. . . . . 19
Using the application debugging profile manager 3270 interface. . . . . 39

**Chapter 2. Starting and stopping CICS 59**

What happens when you start CICS . . . . . 59
The types of CICS startup . . . . . 59
CICS actions on an initial start . . . . . 60
CICS actions on a cold start . . . . . 60
CICS actions on a warm start . . . . . 61
CICS actions on an emergency restart . . . . 64
CICS startup and the z/OS Communications Server session . . . . . 65
End of CICS startup . . . . . 65
Managing CICS shutdown . . . . . 66
The types of CICS shutdown . . . . . 66
What happens during CICS shutdown . . . . 67
Shutting down CICS normally . . . . . 70
Shutting down CICS immediately . . . . . 71
Shutdown command options . . . . . 71
CICS warm-up and cool-down by use of z/OS Workload Manager health service . . . . . 71
Initiating a CICS system warm-up. . . . . 73
Initiating a CICS system cool-down . . . . . 74

**Chapter 3. Administering restart and recovery . . . . . 77**

Units of work . . . . . 77
Shunted units of work. . . . . 77
Locks . . . . . 78
Synchronization points . . . . . 79
CICS recovery manager . . . . . 81
Managing the state of each unit of work. . . . 82
Coordinating updates to local resources . . . . 83
Coordinating updates in distributed units of work . . . . . 84
Resynchronization after system or connection failure . . . . . 85
CICS system log. . . . . 85
Information recorded on the system log . . . . 85
System activity keypoints. . . . . 86
Forward recovery logs. . . . . 86
Replication logs . . . . . 87
User journals and automatic journaling . . . . . 87
Shutdown and restart recovery . . . . . 87
Normal shutdown processing . . . . . 88
Immediate shutdown processing (PERFORM SHUTDOWN IMMEDIATE) . . . . . 91
Shutdown requested by the operating system . . 92
Uncontrolled termination . . . . . 92
The shutdown assist transaction . . . . . 92
Cataloging CICS resources . . . . . 93
Shutdown initiated by CICS log manager . . . 95
How the state of the CICS region is reconstructed 97
Recovery with z/OS Communications Server persistent sessions. . . . . 100
CICS cold start . . . . . 103
Starting CICS with the START=COLD parameter 104
Starting CICS with the START=INITIAL parameter . . . . . 109
CICS warm restart. . . . . 110
Rebuilding the CICS state after a normal shutdown . . . . . 110
Automatic restart management . . . . . 118
CICS ARM processing . . . . . 118
CICS restart JCL and parameters . . . . . 120
Workload policies . . . . . 121
Connecting to the z/OS Communications Server 121
Automatic restart of CICS data-sharing servers 122
Backup-while-open (BWO) . . . . . 124
BWO and concurrent copy . . . . . 124
BWO requirements . . . . . 125
Which data sets are eligible for BWO . . . . 125
How you request BWO . . . . . 126
Removing BWO attributes . . . . . 129
Systems administration . . . . . 129
BWO processing . . . . . 130
An assembler program that calls DFSMS callable services . . . . . 139
Disaster recovery . . . . . 143
Why have a disaster recovery plan? . . . . . 143
Disaster recovery testing . . . . . 144
Six tiers of solutions for off-site recovery . . . 145
Disaster recovery and high availability . . . . 155
Disaster recovery facilities . . . . . 159
CICS emergency restart considerations . . . . 160
Final summary . . . . . 161

**Notices . . . . . 163**

**Index . . . . . 169**


© Copyright IBM Corp. 1989, 2017

# About this PDF

This PDF provides general information about operating CICS, including recovery and restart. Additional PDFs, listed below, provide information about administering certain areas of CICS TS for z/OS. You might also need the companion references to this book: the PDFs called *Supplied Transactions Reference* and the *Utilities Reference*. Before CICS TS V5.4, the information in this PDF was in the *Operations and Utilities Guide* and the *Recovery and Restart Guide*.

Information about administering certain areas of CICS is in the following PDFs:

- Applications is in *Developing CICS Applications*
- Connected CICS systems is in the *Intercommunication Guide*
- Shared data tables is in the *Shared Data Tables Guide*
- Internet connections is in *Internet Guide*.
- ONC/RPC interface is in the *External Interfaces Guide* .
- EXCI is in *Using EXCI with CICS*.
- Front End Programming Interface is in the *Front End Programming Interface User's Guide*.
- DB2 is in *Using DB2 with CICS*.
- DBCTL is in the *IMS DB Control Guide*.
- CICSPlex SM is in the *CICSPlex SM Administration* .

For details of the terms and notation used, see Conventions and terminology used in the CICS documentation in IBM Knowledge Center.

## Date of this PDF

This PDF was created on June 6th 2017.

© Copyright IBM Corp. 1989, 2017

# Chapter 1. Controlling CICS operation

While CICS® is running, you can control its operation by changing CICS system definitions and by deleting and installing resource definitions. Use the CICS Explorer® to control your CICS systems.

### About this task

The CICS Explorer Operations views provide a view of the CICS resources that support day-to-day operation and management of the enterprise. See CICSPlex SM Operations views in the CICS Explorer product documentation.

If you are running your CICS regions in a CICSplex, you can use CICSPlex® System Manager functions to control the operation of CICS. For information, see Introduction to CICSPlex SM.

CICS supplies a number of transactions that you can use to control CICS and its resources while it is running. It also supplies a variety of utility programs, some of which you can use to help with system management.

**Restriction:** Some system attributes defined with system initialization parameters cannot be changed while CICS is running; you must restart CICS with changed system initialization parameters.

## Administering CICS from a console device

You can operate CICS from a console device. A console device can be a locally-attached system console, a TSO user defined as a console, or an automated process such as NetView®.

### Before you begin

To use a console device, you must first define it to CICS. For details, see Defining console devices.

### About this task

You can use the console device for CICS master terminal functions, to control CICS terminals or to control several CICS regions in conjunction with multiregion operation. Normal operating system usage of the console device is not inhibited, and CICS supports multiple console devices where present.

### Procedure

- Define the console device that should be used for putting **MODIFY** commands into your job stream. For more information about putting commands into job streams, see “Using JCL to initiate CICS commands” on page 2.
- Invoke CICS transactions from the console device by using the MVS **MODIFY** command (F for short). Other CICS operators can communicate with the console device operator. Note that the CEDA transaction can be used from a console device to install resource definitions only. The sample programs cannot be executed from a console device.

© Copyright IBM Corp. 1989, 2017

- Use the MVS command `d consoles` to display a list of console devices and their names.
- Use TSO CLIST processing, or an automated process such as NetView, to issue sequences of CICS commands. To associate command responses with the originating command from an automated process, add a command and response token (CART) to the originating command. CICS returns this CART in all write-to-operator (WTO and WTOR) macros issued in response to the command.

## Entering commands from a console device

You can invoke CICS transactions from the console device by using the MVS **MODIFY** command (F for short).

To enter a CICS command from a console device, use:

```
{MODIFY|F}  cicsid,[']command[']
```

where:

**cicsid** is the region identifier for the CICS region. This is one of the following:

- name of the job being used to execute CICS
- name of a procedure if CICS was initiated as a started task without a qualifier
- name of the task identifier qualifier if CICS was started as a started task with a qualifier.

**command**
is a string of data, starting with a CICS transaction identifier.

**Example:**

```
MODIFY DFHIVPOL,'CEMT INQUIRE TASK'
```

If a transaction started at a console device requires further input, you are prompted in the same way as a terminal operator. For more information about continuing transaction input, see "Replying to messages from transactions started at console devices" on page 5.

## Entering commands from TSO

A TSO user can enter CICS commands after invoking the TSO command **CONSOLE**.

Enter the CICS commands in either of the following formats:

- `CONSOLE SYSCMD ({MODIFY|F}  cicsid,[']command['])`
- ```
  CONSOLE
  {MODIFY|F}  cicsid,[']command[']
  END
  ```

When the TSO command **CONSOLE** is used, TSO checks the user for authority to issue console commands. Further, if console operator command security is active, the TSO user must be specifically authorized to issue `MODIFY cicsid`.

## Using JCL to initiate CICS commands

If you have defined a console entry in your CSD as CONSNAME(INTERNAL), you can submit commands to your CICS region by using JCL.
  ```

### About this task

Edit your JCL as follows. The normal rules of JCL apply.

### Procedure

1. Edit your JCL to use the MVS command **MODIFY**.
2. Follow the MODIFY command with the job name or task ID of the CICS region you are addressing, followed by the CICS commands.
3. Submit the JCL. The following sample job shows how you might submit commands in this way.

```
//IEFBR14 JOB (accounting information),CLASS=A,MSGCLASS=A,MSGLEVEL=1,...,...
//*
//*  Sample JOB to submit CICS commands using CONSNAME(INTERNAL)
//*
//IEFBR     EXEC PGM=IEFBR14
// F CICSRUN,'CEMT INQ TER'
// F CICSRUN,'CEMT INQ TAS'
// F CICSRUN,'CEMT SET TER(L77C) ACQ'
//
```

### Results

If you omit the apostrophes around the CICS command, and there are sequence numbers at the end of the line, the numbers are passed to CICS as part of the command. This causes CICS to display a warning message on the console, but the command is still obeyed.

## Console device messages

During both the initialization and the running of CICS, various messages appear on your console device. These are mainly for information, but in some cases could require a reply or some action from you.

Console messages might be subject to message formatting if you have defined CICS as an MVS™ subsystem with console message-handling support. The term **console message** is used for messages sent to the system console, and does not refer to CSMT messages or the JES joblog.

### Suppressing information-only messages

You can use the system initialization parameter **MSGLVL** to control the generation of messages to console devices. If you code MSGLVL=0, only critical errors or interactive messages are printed.

### Sample console messages

Sample console messages issued when CICS starts up are given in System console messages for CICS startup. Sample console messages issued when CICS shuts down are given in Managing CICS shutdown.

### Console message-formatting

The main purpose of the console message-handling facility is to ensure that all messages issued by CICS regions contain the APPLID of the CICS region issuing the message.

By using this facility, CICS can enable MVS to:

- Convert all console messages to the same format, and

- Inserts the applid of the sending region into each message.

You specify that CICS is to use the console message-handling facility when you define CICS as an MVS subsystem (by the CICS entry in the IEFSSNaa member of the SYS1.PARMLIB library). If the message-handling facility has been defined for CICS, all messages from all CICS regions of any release are intercepted and reformatted, if necessary, to include the APPLID, provided that at least one CICS region is running in the MVS image.

For information about defining CICS as an MVS subsystem with support for the console message-handling facility, and about activating the facility, see Defining CICS as an MVS subsystem.

**Message format:**

The following examples show three messages as they appear with and without console message formatting. The examples use CICSIDC as the applid of the sending region.

- Message format without console message formatting:

```
DFH5730 - USER RECOVERY BEGINNING
DFH5731 - NO ACTIVE USER RECORDS ON THE SYSTEM LOG
DFH5732 - USER RECOVERY COMPLETED
```

- Message format with console message formatting:

```
DFH5730 CICSIDC  USER RECOVERY BEGINNING
DFH5731 CICSIDC  NO ACTIVE USER RECORDS ON THE SYSTEM LOG
DFH5732 CICSIDC  USER RECOVERY COMPLETED
```

**Advantages of message formatting:**

The main benefits of using console message formatting include assistance to the console operator and ease of automated operation by a program such as NetView.

The implementation of message formatting also:

- Allows masking of the password entered at the console during the CICS signon transaction. For example, you might enter the following command to sign on to CICS from a console:

  ```
  F CICS,CESN USERID=HARBEN, PS=HUMMER, NEWPS=STONE
  ```

  The passwords are then obliterated with asterisks when the command is redisplayed on the console or recorded in the system log.

  ```
  F CICS,CESN USERID=HARBEN, PS=********, NEWPS=********
  ```

- Allows the adding of a set of MVS generic routecodes to all CICS console messages, permitting them to be sent to a defined set of consoles.
- Removes the restriction that prevents the use of the name *CICS* as the MVS jobname of a CICS region that is started with the START command.

## Replying to messages

If one or more CICS messages are followed by an associated message that requests an operator response, the earlier message or messages might have scrolled off the console screen before the response-requesting message appears.

### About this task

Some messages that need a reply include a preceding message number or specify a response that can be entered to display the preceding message.

If a message requests a reply but does not provide means of determining the previous messages that explain the response required, CICS retains all messages in the logically-related set in the message buffer, until a valid response is received to the final message. When the console displays a message that requires a response, the operator can request a display of all preceding related messages. A typical message that needs a response is:

```
DFHSI1552 applid Restart error reported above. Reply 'GO' or 'CANCEL'.
```

If such a message appears, the operator can display all the preceding related messages by entering the MVS command:

```
DISPLAY R,I
```

When a valid response is received to the final message in the set, CICS deletes all the related messages from the message buffer.

## Replying to messages from transactions started at console devices

If a transaction started at a console device requires further input, you are prompted in the same way as any normal terminal operator.

### About this task

You can continue the input in one of the following ways:

- If the transaction is conversational and uses the **CONVERSE** or **RECEIVE** command, the message from CICS will contain a reply number that must be quoted in the reply. This is described in this section.
- If the transaction is pseudo-conversational, you must enter further **MODIFY** commands to continue the conversation.

You respond to messages from transactions started at a console device by using the REPLY command (abbreviation R). For example:

```
REPLY 02,'datastring'
```

where 02 is the number of the message to which you are replying, and ‘datastring' is your reply. If you cancel a transaction that is running at a console device, and the transaction is awaiting a reply, the outstanding reply is also canceled.

For information about using CEMT and the other CICS-provided transactions, and about entering transactions from a console, see .

If you try to communicate with an active CICS region from a console device that has not been defined to CICS, you get message DFHAC2015 saying that your console has not been defined to CICS and that your input will be ignored.

In a CICS region that has consoles and SNA LUs, a console can remain active when CICS and the z/OS® Communications Server for SNA are disconnected from each other. This means that you can use the console to open or close the CICS-z/OS Communications Server connection without CICS being terminated.

## Suppressing and rerouting messages

CICS provides a global user exit point, XMEOUT, that is invoked before a message is sent from the message domain to its destination. XMEOUT can be used to invoke an exit program to intercept messages issued by SEND MESSAGE requests, and suppress the messages, change their destination, or leave them alone.

### About this task

CICS provides six sample user exit programs, DFH$SXP1 through DFH$SXP6, which you can use to suppress or reroute messages.

For programming information about this global user exit and the sample user exit programs, see XMEOUT and the user exit programming interface (XPI)

## Administering with CICS supplied transactions

CICS provides operations, usually initiated from terminals, called *transactions*, each of which involves the use of CICS tables, programs, and internal services. The following sections describe the transactions that are supplied by CICS and that have an operator interface.

CICS transactions have identification codes that start with "C" and are 4 characters long; for example, CEMT. For a complete list of CICS transactions, including those that do not have an operator interface, see the List of CICS transactions.

In general, you start a CICS transaction by entering its transaction identifier; for example, CEMT. The transaction identifier is used by CICS to identify the programs that handle the specified transactions, and to establish a task to process them.

If you use an IBM® 3270 system display or similar display device that has the appropriate features installed, you can also start a transaction by a program function (PF) key or program attention (PA) key, by an operator identification card reader, by a magnetic slot reader, or by a light pen.

You might want to apply a CICS-supplied upgrade, but are using modified versions of one or more CICS-supplied transactions or of the CICS-supplied calling programs that handle CICS-supplied transactions. After you have first copied them to differently named groups, you must replace these private versions from the upgraded CICS-supplied version and modify them afresh to ensure that the necessary upgrade changes are carried out. Failure to do this can lead to unpredictable results.

This section describes:


- "Starting or stopping a transaction"
- "Syntax notation and conventions used" on page 7
- "Terminal operation" on page 9
- "Using the system console" on page 14
- "Using TSO consoles" on page 16


### Starting or stopping a transaction

You start a CICS transaction by pressing the CLEAR key to clear the screen, and entering the transaction identifier, either by itself or followed by data, on the command line of the screen. The command line is a single line, usually at the top of the screen.

#### About this task

You can type the transaction identifier by itself and follow the prompts until a complete transaction command is built up, or you can type the complete

transaction command on the command line. If you do not enter enough information, or if the information you enter is wrong, you are prompted to complete or correct your input.

### Example

For example, in the following transaction, CEMT is the transaction identifier and the additional data is INQUIRE PROGRAM(PROG1).

```
CEMT INQUIRE PROGRAM(PROG1)
```

When the transaction starts, it processes the additional data. At the completion of this transaction, you get the following message:

```
STATUS: SESSION ENDED
```

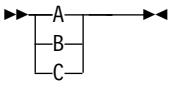
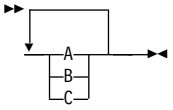
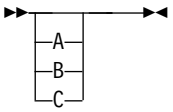
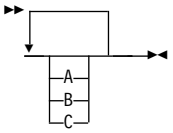
### What to do next

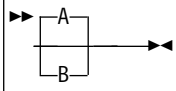
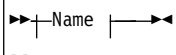
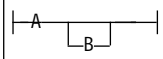
After a transaction has completed, press the CLEAR key to clear the screen in readiness for the next transaction. You can cancel any request by typing CANCEL on the command line.

## Syntax notation and conventions used

Each command has a syntax box to show you what options there are. You interpret the syntax by following the arrows from left to right.

The conventions are:

| Symbol | Action |
|---|---|
|  | A set of alternatives—one of which you ***must*** code. |
|  | A set of alternatives—one of which you ***must*** code. You ***can*** code more than one of them, in any sequence. |
|  | A set of alternatives—one of which you ***can*** code. |
|  | A set of alternatives — any number (including none) of which you can code once, in any sequence. |

| Symbol | Action |
|---|---|
|   | Alternatives where **A** is the default. |
|   **Name:**   | Use with the named section in place of its name. |
| Punctuation and uppercase characters | Code exactly as shown. |
| Lowercase characters | Code your own text, as appropriate (for example, `name`). |

## Minimum abbreviation of keywords

In general, the CICS transactions accept as few characters of a keyword as needed to identify it uniquely in the request.

For example, in a CEMT INQUIRE TASK command, you can use TASK, TAS, or TA to uniquely identify TASK. You cannot use T alone because that could be confused with TCLASS, TERMINAL, TRACE, or TRANSACTION.

In the syntax displays on your screen, the minimum permitted abbreviation is shown in uppercase characters, and the remainder is shown in lowercase.

Minimum abbreviations might change between CICS releases because of the introduction of new commands.

## Uppercase input to transactions

In general, most CICS-supplied transactions accept only uppercase input. If UCTRAN=YES has been specified in the terminal definition, all lowercase characters, even those enclosed within single quotation marks, are translated to uppercase.

If you have to specify UCTRAN=NO for your terminal, you have to ensure that the group specified for your terminal refers to a profile that will carry out uppercase translation.

CICS provides a PROFILE definition, DFHCICSP, in the DFHSTAND group in the CICS system definition (CSD) file. This profile is identical to DFHCICST except that it specifies UCTRAN(YES) instead of UCTRAN(NO).

The new profile is used by the CICS-supplied page retrieval transaction, CSPG. The new profile, together with changes in the task-attach routine and the page retrieval program, enables CICS to perform uppercase translation at the transaction level for BMS paging.

This allows users of terminals that are defined with uppercase translation switched off to use the page retrieval function without having to enter paging commands in

upper case. Assigning a new profile for CSPG means that all data entered on the retrieval command (defined by the PGRET system initialization parameter) and the purge command (defined by the PGPURGE system initialization parameter) is translated to uppercase.

If a user's terminal is defined with UCTRAN(YES), the new profile has no effect because all terminal input is translated to uppercase anyway.

## Terminal operation

A CICS system makes provision for the following classes of operators; terminal operator, a supervisory terminal operator, and a master terminal operator.

- A *terminal operator* who can use a terminal to perform routine transactions that cause application programs to be processed. You can use a small selection of CICS transactions. For example, you can inquire about, or change, the status of your own terminal.
- A *supervisory terminal operator* who can perform all the duties of a terminal operator, in addition to supervising other operators within a functional group. Your operator security code gives you access to the supervisory terminal transaction, CEST, with which you can monitor and control some of the system resources used by your group.
- A *master terminal operator* who can monitor and control resources in a CICS system. Your operator security code gives you access to the master terminal transaction, CEMT, with which you monitor and control the system resources. Internal security checking might limit the range of resources under your control.

### $ (the dollar symbol)

In the character sets given in this information, the dollar symbol ($) is used as a national currency symbol and is assumed to be assigned the EBCDIC code point X'5B'.

In some countries a different currency symbol, for example the pound symbol (£), or the yen symbol (¥), is assigned the same EBCDIC code point. In these countries, the appropriate currency symbol should be used instead of the dollar symbol.

### Operator security

The transactions you can initiate are defined by your profile in the external security manager (ESM) database, which is normally provided when you sign on using the CESN transaction.

Generally, the master terminal operator has access to all CICS-supplied transactions, the supervisory terminal operator has access to a subset, and the terminal operator has access to very few transactions.

The system programmer is responsible for allocating operator security codes to restrict the use of particular transactions. For more information, see the relevant system administration guide for the ESM you are using. For example, if you are using RACF, see the *z/OS Security Server RACF Security Administrator's Guide*.

### Terminal operator

To operate the system, you normally first sign on to the system and, as a minimum, enter your user ID and your password.

During signon, the information you enter is used by CICS to establish priorities and your ESM profile for the transactions that you might want to use later. When you have signed on, you have access to those transactions defined in your ESM profile.

After you have signed on, you can enter only specific transaction identifiers. Be aware of error messages that might be generated by the transactions you start, and the corrective action that you must take. In addition to error messages, be aware of other messages that CICS might transmit to your terminal.

You need to know the terminal identifiers of other terminals with which you want to communicate.

## Supervisory terminal operator

A supervisory terminal operator is the supervisor of any part of the system for which group control is required. You are responsible for supervising, and keeping operational, groups of terminals defined in one or more terminal list tables (TLTs). You do this using the supervisory terminal transaction, CEST.

Your responsibilities can be thought of as a subset of those of the master terminal operator, and you should be aware of those functions that are not available to the terminals under your supervision. You also should be aware of, and understand, the procedure for changing the status of each terminal.

You should know the identifiers of all terminals and operators under your supervision. The terminal identifiers are defined in one or more CICS terminal list tables (TLTs). Individual TLTs can be identified by a 1-or 2-character suffix you enter as the SUPRID and CLASS(value) keywords of the CEST transaction.

When you use the CEST command for *all* terminals defined in a TLT, you have to specify the SUPRID keyword followed by the 2-character identifier of that TLT.

A subset of those terminals can be grouped together as a *class*, and can be defined as such in a different TLT. For information about defining the TLT itself, see TLT - terminal list table . You can specify that class by means of the CLASS(value) keyword of the CEST SET TERMINAL command, where ‘value’ is the suffix that identifies the TLT in which the class of terminals has been defined.

Alternatively, you can name one or more terminals in the CEST SET TERMINAL(value) command itself.

If you frequently want to restrict a CEST command to a subset of your terminals, and have defined another TLT identifying that subset, you then have to use the CEST SET TERMINAL SUPRID(value) keyword to refer to the “main” TLT, followed by the CLASS(value) keyword to refer to the specific TLT containing the subset.

So, for example, if you have defined terminals S202, S203, S204, and S205 in DFHTLTAB and you want to issue a CEST command that sets *all* those terminals out of service, you issue the following command:

```
CEST SET TERMINAL SUPRID(AB) OUTSERVICE
```

If, on the other hand, you want to restrict your command(s) to terminals S202 and S204, for example, you could define these two terminals in another TLT - DFHTLTCD, say - and issue the following command:

```
CEST SET TERMINAL SUPRID(AB) CLASS(CD) OUTSERVICE
```

Alternatively, of course, you could issue the following command:

```
CEST SET TERMINAL(S202,S204) SUPRID(AB) OUTSERVICE
```

Unless otherwise stated, the information in this book about the supervisory terminal and the CEST transaction applies only to a single CICS system, regardless of whether it is connected to another CICS system through ISC or MRO.

## Master terminal operator

The master terminal operator controls system components using the master terminal transaction, CEMT. With this transaction, the master terminal operator can dynamically change the system control parameters.

Although the transaction can be started at any valid IBM 3270 family display device or equivalent terminal, or from the operating system console, its use is intended to be limited to a person known as the *master terminal operator*. Starting a transaction from the operating system console is described in "Using the system console" on page 14.

The control permitted through CEMT allows you, the master terminal operator, to improve performance by changing the system control parameters in the day-to-day operation of the system. In addition to system control, you have prime responsibility for administering the terminal facilities of the system.

By using the routing transaction (CRTE), you can also be a master terminal operator for multiple connected CICS systems.

As the master terminal operator, you can access all terminal and supervisory terminal transactions. In addition, however, you must be familiar with all the procedures associated exclusively with the master terminal. You must be aware of which terminals and operators can access CICS at any given time, and of the identifiers by which they are known to CICS.

For example, when inquiring about terminals, you can specify a class of terminals or a list of terminals. A class of terminals is specified by the CLASS(value) keyword, where 'value' is the 1-or 2-character suffix of the related terminal list table (DFHTLTxx). A list of terminals is specified by a series of terminal identifiers following the CEMT SET TERMINAL(t1,t2,...,) command, where t1, t2, are terminal identifiers. See List of resource identifiers for more information.

For MRO and LUTYPE6.1 connections, you must know the identifier of each parallel session, and specify this identifier when operating on the session.

For LUTYPE6.2 (APPC) connections, you must know the modename of each set of parallel sessions, and specify this modename when operating on the modegroup.

Your use of the master terminal transaction is restricted by entries in the signon table and in the installed transaction definitions. These entries are the responsibility of the system programmer.

During long periods of continuous operation, you can, at intervals, read out and reset the statistics counters. The volume of activity in your system determines how often you should do this.

When the system has satisfactorily completed its response to a command, the time and date are printed or displayed at your terminal, as follows:

```
TIME=hh.mm.ss DATE=mm.dd.yy
```

where time is in hours, minutes, and seconds, and date is in months, days, and years, or in the form specified by the DATFORM system initialization parameter. For brevity this final message has been deleted from all further examples.

Unless otherwise stated, the information about the master terminal and its transactions applies only to a single CICS system, regardless of whether it is connected to another CICS system through ISC or MRO.

## MRO and ISC support

Multiregion operation (MRO) and intersystem communication (ISC) allow the sharing of resources between more than one CICS region. Thus a user at a terminal assigned to one CICS region can run transactions in connected regions, and can access resources - files, for example - in other regions.

It is also possible for a transaction running in one region to communicate with a transaction running in another region, thus sharing the processing workload.

Except for experiencing longer response times, you should not be aware that MRO or ISC processes are being used.

## BMS partitions

When you use display devices that support BMS partitions, make sure that you understand how to use the SCROLL, PARTITION JUMP, CLEAR, and CLEAR PARTITION keys, the concept of the active partition and the meanings of the partition-related indicator symbols that can appear on a display screen.

For information about BMS partitions, see Basic mapping support.

### CLEAR key

The CLEAR key clears all partitions from the display, and sets the terminal to 'base' state. The next BMS output request re-creates the partitions (but does not restore their contents), using the application partition set.

The CLEAR and CLEAR PARTITION keys cannot be used interchangeably when an existing CICS transaction is run in a single explicit partition.

### Partitions and the execution diagnostic facility

The execution diagnostic facility (EDF), invoked by CEDF, is unavailable in single-screen mode on a terminal in partitioned state. EDF must be used in dual-screen mode for debugging application programs that use partitions.

### Partitions and the command interpreter

The CICS command interpreter, invoked by CECI or CECS, cannot be used to process commands that refer to partitions. This is because the command interpreter display cannot be restored after the screen has been partitioned.

### PA1 print key

The PA1 print key is not supported from a terminal in partitioned state.

### Routing and multiple partitions

Routed messages can be directed to a terminal, including the transaction terminal, which supports partitions. However, such messages reset the terminal to 'base' state.

### Terminal paging

When a BMS logical message is saved in CICS temporary storage, CICS also saves the application partition set. This partition set is loaded onto the target terminal before any pages are delivered. CICS builds a separate page for each partition, and overflow occurs on a partition basis.

### Page retrieval

Terminal-operator page-retrieval commands operate on a partition basis. When a page-retrieval or page-copy command is entered in a partition, it implicitly refers to pages in that partition. If single-keystroke retrieval is used, the retrieval command applies to the partition containing the cursor when the PF key is pressed. The first page for a partition is displayed initially in the viewport.

### Message chaining

CICS retains a current partition for each level of page chaining. This is initially the default partition for that partition set. Page-retrieval commands entered on a cleared screen, or page-retrieval commands for a chaining level other than the one being displayed, refer to the current partition for the target chaining level. The current partition is reset to the partition in which the last terminal-operator command was entered.

CICS retains the current page for each partition in the partition set. This is initially the first page. The current page is redisplayed in each partition in the following circumstances:

- For the initial display when the BMS paging program is first invoked
- Following erasure of the terminal partition set caused by pressing the CLEAR key
- Following page retrieval for a different page-chaining level
- Following page purge for a different page-chaining level.

### Copying pages

BMS page copy operates on a partition basis (not a screen or partition set basis). BMS page copy copies a page from a partition to any terminal in 'base' state. You cannot copy a page from a partition to another partition on the same or another terminal.

### Message termination

When you terminate a message, the entire logical message (that is, all pages in all partitions) is purged, irrespective of the partition in which you entered the purge command. The response to a page query request is displayed on a cleared, unpartitioned screen.

### Error messages

Most error messages relating to invalid paging commands are displayed with an erase or write in the partition in which you entered the command. Other error messages unrelated to any particular partition (such as those relating to invalid message identifiers) are displayed on a cleared unpartitioned screen.

## Using the system console

Console support makes it possible for a terminal to be both an operating system console and a CICS master terminal.

### About this task

If multiple console support (MCS) is in use, you can define each console to CICS as a separate terminal, and all consoles can communicate with CICS simultaneously.

You can use any operating system console as a CICS terminal, if it has been specified on the CONSOLE keyword of the **CEDA DEFINE TYPETERM** command. If this has not been done, you get the following message when you try to use the console:

```
DFHAC2015 This console has not been defined to CICS.
```

and your input is ignored.

All consoles that have been defined as CICS terminals can use automatic transaction initiation (ATI), and can receive messages from other terminals and consoles, as well as from CICS transactions.

In a system that has consoles and SNA LUs, a console can remain active when CICS and the z/OS Communications Server are disconnected from each other. You can use the console to make or break the CICS-z/OS Communications Server connection without CICS being terminated.

Use the MODIFY and REPLY commands to start the CICS-supplied transactions from an operating system console.

In addition to the MODIFY and REPLY commands, the system programmer should consider use of the CONTROL, DISPLAY, START, and VARY commands when preparing console operator procedures. For information on these commands and other system details, see the *z/OS MVS System Commands* manual.

### Rules for console entry

Commands typed at a console are translated to uppercase, except for characters enclosed within single quotation marks (' '), which remain unchanged. The occurrence of a literal single quotation mark must be indicated by a pair of single quotation marks (''), for example:

```
'Please phone Mr O''Neill'.
```

If UCTRAN=YES has been specified in the terminal definition, all lowercase characters, even those enclosed within single quotation marks, are translated to uppercase.

## MODIFY command

You start a CICS transaction from a console by using the MODIFY command, as follows:

```
MODIFY ident,datastring
```

You can abbreviate the MODIFY command to F.

`ident` can be any of the following:

- The name of the job used to start CICS, when it is started by a job stream.
- The name of the procedure used to start CICS, when it is started by an MVS START command, for example:

  ```
  START procedure_name
  ```

  where "procedure_name" is the ident value.
- The task identifier that was used to qualify the procedure name, for example:

  ```
  START procedure_name.taskid
  ```

  where "taskid" is the ident value. This is likely to be used where the same procedure is started more than once.

`datastring` is a string of data, starting with a CICS transaction identifier.

For example, to start transaction CEBT on the CICSA system from the console, type:

```
MODIFY CICSA,CEBT PERFORM TAKEOVER
```

You can type more than one MODIFY command at a console; each is processed in order of entry.

A CICS transaction can issue terminal control READ, WRITE, or CONVERSE commands to communicate with a console operator. WRITE and CONVERSE transmit application program messages, but READ produces a prompt, incorporating message 'DFH4200A', as follows:

```
@nn DFH4200A jjjjjjjj tttt
```

where:

**nn** is the number (generated by the operating system) that you must use in your reply to the prompt. Messages from a transaction that uses CONVERSE commands also contain this number.

**jjjjjjjj** is the jobname of CICS in the operating system.

**tttt** is the transaction identifier of the CICS transaction that has issued the READ command.

## REPLY command

You (the console operator) must respond to each prompt by using the REPLY command, which you can type at either the prompted console or the master console:

```
R[EPLY] nn,datastring
```

where `nn` is the number of the prompt to which you are replying, and `datastring` is your reply.

If a transaction is purged while it is awaiting a reply from the operator, the reply is canceled.

You should note that messages to the console can become interspersed with messages from the operating system and from other regions, making them difficult to read. In extreme cases, parts of lengthy messages can 'scroll off' the console screen before they have been read.

### Example of a conversation using CONVERSE

```
modify job002,serv  1

@17 FAULT TYPE?   2

r 17,elec   3

MESSAGE HAS BEEN SENT
```

1 MODIFY command specifying that transaction "serv" is to be started; this transaction sends messages to service groups supporting the installation.

2 The transaction response produced by a CONVERSE command and relayed by the operating system.

3 Your reply that the fault is an electrical one.

### Example of a conversation using WRITE/READ

```
modify job002,usid   1

USER SIGNON ID=?   2

@25 DFH4200A JOB002 USID  3

r 25,accts1   4

USER'S NAME: J. SMITH   5
USER'S TEL. NO.: 88999   6
```

1 MODIFY command specifying that transaction "usid" is to be started. This transaction provides information about the user identified by "usid".

2 Application-program message produced by a WRITE command.

3 System message produced by a READ command.

4 Your reply.

5 Transaction message, produced by a WRITE command, giving the requested information. No reply is needed.

6 Another transaction message, produced by a WRITE command, giving more requested information. Again, no reply is needed.

## Using TSO consoles

A TSO session can be used to input CICS commands. This has several advantages as it removes the MVS limitation of 99 consoles, it supports additional device types such as the IBM 3290, and it supports remote operation.

### About this task

The console ID is 4 bytes, only one of which is used for locally connected consoles. TSO and JES3 consoles use all 4 bytes. In addition, each console in a sysplex has an 8-byte name.

## Using the KILL option for purging transactions

It is possible for a looping CICS task or shortage of CICS resources to bring all normal processing in a CICS region to a halt. The KILL option helps you to remove such tasks more quickly from a CICS region, with the minimum impact on the integrity of your systems.

### About this task

You must use the KILL option only when it becomes impossible for you to cancel the task or tasks by using any of the standard CICS mechanisms. The KILL option does not guarantee integrity of any kind, but in some situations allows you to free a stalled region, enabling the region to continue processing.

### Procedure

1. Try to cancel the task using the PURGE or FORCEPURGE option on the **CEMT SET TASK** command. Both PURGE and FORCEPURGE might enable you to remove the task, retaining system and data integrity. If neither PURGE or FORCEPURGE respond within a reasonable time, use the KILL option.
2. Use the KILL option on the appropriate command, depending on what task you are trying to cancel. The KILL option is available on **CEMT SET TASK**, **CEMT SET CONNECTION**, and **CEMT SET TERMINAL** commands.
3. If you cannot list the active tasks in the system to determine which task or tasks should be canceled, use the CEKL transaction.
   a. Enter the command `CEKL INQUIRE TASK` to list information about tasks in the CICS region; for example, all tasks that are suspended.
   b. Enter the command `CEKL SET TASK` to cancel a selected task using the PURGE, FORCEPURGE or KILL option.
4. Having identified the problem task, you can then use the normal process of attempting to remove the task from your system using either PURGE or FORCEPURGE through the CEMT SET TASK (value) PURGE transaction. Initially you can track the state of the task to see if it has successfully removed. Two things might have happened:
   a. The task was not purged within an acceptable length of time.
   b. The CEMT transaction did not respond.

   If CEMT is unavailable, consider using the CEKL command outlined in step 5.

   **Important:**

   If the task could not be purged but CEMT is available, see step 6 before trying to using the KILL option. You must try FORCEPURGE before using KILL.
5. You can identify problems in your system using the CEMT transaction or CICSPLEX SM. However, if these are unavailable or not running (because the QR TCB is looping or suspended), you can use the new command, CEKL. You can use the CEKL command to do the following:
   a. To identify the problem tasks.

b. To remove the tasks by executing the PURGE, FORCEPURGE, or KILL option.
c. To monitor the effects of attempting to remove the tasks from your system.

For full details about CEKL, see CEKL.

6. If you use the KILL option, an abend code is issued that reports the protective state of the task (purge or forcepurge protection) when it was killed. Regardless of the state of the transaction KILL cannot guarantee any data or system integrity.

**DANGER**

**This action can result in unpredictable effects, including overwriting of data within the CICS region or abnormal termination of the CICS region. It should only be used as a last resort.**

Using CEMT (or equivalent SPI commands), you are not allowed to kill a task without having first attempted to forcepurge it. You can use CEKL to kill the task without having first forcepurged it. Use CEKL is very much the option of last resort.

Removing tasks can have an effect on the integrity of your system. The following table shows the possible consequences of using purge, forcepurge, or kill.

*Table 1. Data and system integrity*

| Command | System integrity | Data integrity |
|---|---|---|
| PURGE | YES | YES |
| FORCEPURGE | YES | NO |
| KILL | NO | NO |

CICS ignores a request to purge a task if the task is marked non-purgeable. CICS might ignore or delay processing a request because of a risk of damaging CICS system integrity, for example, because I/O might still be in progress to a buffer owned by the task, or because recovery processing would be unable to complete.

## Administering with the application debugging profile manager

Use the application debugging profile manager to manage your debugging profiles.

You can use it to perform the following functions:

- Display a list of debugging profiles
- View the contents of profiles
- Create new profiles
- Modify existing profiles
- Copying existing profiles
- Delete debugging profiles
- Activate and inactivate profiles
- Associate a debugging display device with active profiles

The application debugging profile manager has two user interfaces:

- A web browser interface. For more information, see “Using the application debugging profile manager Web interface” on page 19.

- A terminal interface (the CADP transaction). For more information, see “Using the application debugging profile manager 3270 interface” on page 39.

A utility transaction (CIDP) lets you inactivate all debugging profiles in the system. For details, see CIDP - inactivate debugging profiles.

## Using the application debugging profile manager Web interface

Start the application debugging profile manager by typing its URL in your Web browser. The URL that you enter will depend upon how CICS Web support is configured. Your system administrator will tell you the URL to use for your system. CICS displays the “List profiles” page.

### About this task

For example, if your Web browser connects directly to CICS, and your system is configured to use the sample analyzer program DFHWBADX, the URL is:

`http://mvs_address:port/CICS/CWBA/dfhdpwb`

If you are using URIMAP definitions to manage requests, the URL is as defined in the URIMAP definition that references program DFHDPWB.

### Configuring access to the application debugging profile manager Web interface

To provide access to the Web interface, you can use either an analyzer program, or URIMAP definitions, as part of your CICS Web support architecture.

#### About this task

s explains the architecture elements that are used to give Web clients access to CICS applications.

Only one URL is required. The user accesses different pages by selecting interface elements on the pages that the program provides, beginning with the “List profiles” page.

#### Procedure

- If you want to use the sample analyzer program DFHWBADX, or an analyzer program with similar function, to handle requests for the application debugging profile manager, configure the path component of the URL to give CICS the information it needs to run the program. The path `/CICS/CWBA/dfhdpwb` tells the analyzer that:
  - No converter program is required (`CICS` indicates this).
  - The default transaction ID CWBA is used.
  - The program DFHDPWB, which is the application debugging profile manager, is to be run.
- If you want to use URIMAP definitions to handle requests for the application debugging profile manager, you need to set up two URIMAP definitions.
  - One URIMAP definition matches the URL that is entered by the user to start the application debugging profile manager's Web interface, and maps to the program DFHDPWB.
  - The second URIMAP definition is used to run the CICS program DFHADWB1, which displays graphics.

The second URIMAP definition is required because when a graphic is wanted, the HTML pages for the Web interface use the tag `img src=`, so that the Web browser makes a request for the graphic and then displays what it receives. The `img src=` tag specifies the name of the graphics program, and the Web browser makes the request using the path component of the URL that was originally used to start the Web interface, followed by the name of the graphics program. A URIMAP definition is needed to map this new request to the graphics program.

### Example

The following sample URIMAP definitions could be used to provide access to DFHDPWB and DFHADWB1:

```
For DFHDPWB:

URIMAP:       ADPM                - URIMAP name
GROUP:        MYGROUP             - any suitable
USAGE:        SERVER              - For inbound requests
SCHEME:       HTTP                - Or use HTTPS for SSL
HOST:         *                   - * matches any host name
PATH:         /cicsapps/adpm/*    - Any path can be used, with final asterisk
TCPIPSERVICE:                     - If blank, applies to all ports
ANALYZER:     NO                  - Means do not run analyzer
CONVERTER:    CICS                - Means no converter used
TRANSACTION:  CWBA                - Default transaction ID
PROGRAM:      DFHDPWB             - App debugging profile manager

For DFHADWB1:

URIMAP:       ADWB1
GROUP:        MYGROUP
USAGE:        SERVER
SCHEME:       HTTP
HOST:         *
PATH:         /cicsapps/adpm/dfhadwb1
[Same path as for DFHDPWB, but with dfhadwb1 appended]
TCPIPSERVICE:
ANALYZER:     NO
CONVERTER:    CICS
TRANSACTION:  CWBA
PROGRAM:      DFHADWB1             - Graphics program
```

When URIMAP definitions are used, the path specified in the URL does not need to have any relationship to the resource; the linkage between the path and the resource is made by the URIMAP definition. For the URIMAP that points to DFHDPWB, you could specify a path that includes a page element, such as `/cicsapps/adpm/start.html`. The name of the page element does not matter. Note that if you do this, when the Web browser makes a request for DFHADWB1, it omits the page element and appends `dfhadwb1` in its place. If the path in the first of the sample URIMAP definitions was `/cicsapps/adpm/start.html`, the path in the second sample URIMAP definition would still be `/cicsapps/adpm/dfhadwb1`.

.

## The List profiles page

Use the “List profiles” page to display a list of debugging profiles:

- When you start the application debugging profile manager's Web interface, CICS displays the “List profiles” page.

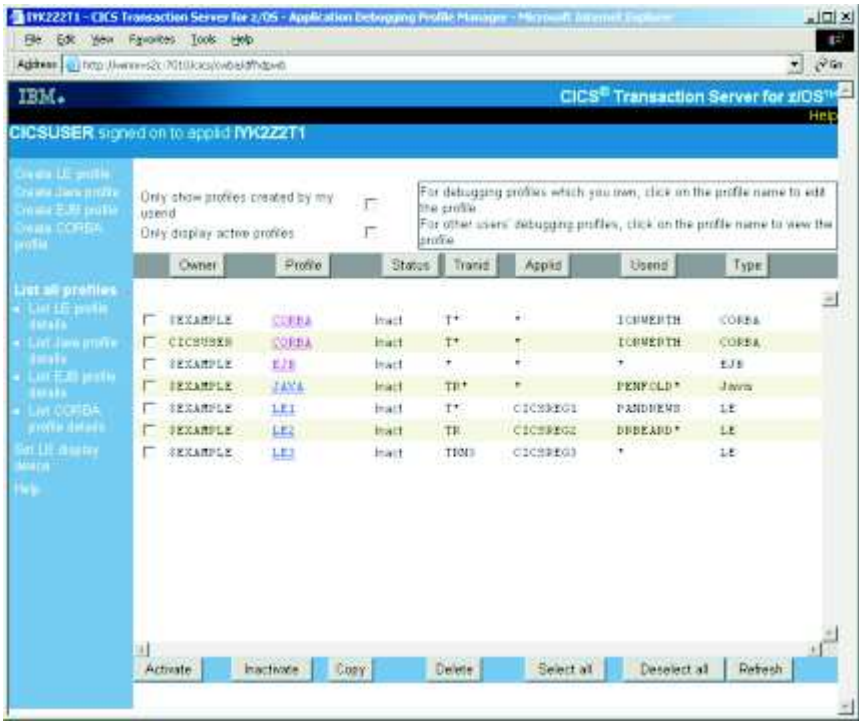



*Figure 1. The List profiles page*

When you use the debugging profile manager for the first time, CICS displays all profiles. Subsequently, CICS displays the profiles that were selected when you last used it.

If there are more profiles than can be displayed in the window, use the scrollbars to scroll backwards and forwards through the list. If you have no profiles, CICS displays an empty list.

There are four variants of the "List profiles" page:

**List LE profiles**
Lists only the profiles for compiled language profiles

**List Java™ profiles**
Lists only the profiles for Java programs

**List EJB profiles**
Lists only the profiles for enterprise beans

**List CORBA profiles**
Lists only the profiles for stateless CORBA objects

The behavior of these pages is identical to the "List profiles" page; however the information displayed on each is specific to the type of profile.

The list contains selected information from the debugging profile. The columns on the page are:

**Owner**
The userid of the profile owner; that is, of the user who created the profile.

**Profile**
The name of the profile

**Status** The status of the profile (**Act** for Active, or **Inact** for Inactive)

The following columns display information specified when the profile is created:

**Tranid** Displays the contents of the **transaction** field

**Program**
On the "List profiles" and "List LE profiles" pages only , displays the contents of the **program** field.

**Compile Unit**
On the "List profiles" and "List LE profiles" pages only, displays the contents of **Compile Unit** field.

If the Compile Unit name is too long to display in the available space, the leading characters are displayed, followed by "**...**". To display the Compile Unit name in full, click the profile name.

**Applid**
Displays the contents of the **Applid** field

**Userid**
Displays the contents of the **Userid** field

**Termid**
On the "List profiles" and "List LE profiles" pages only, displays the contents of the **Terminal** field.

**Type** On the "List profiles" page only, displays the type of program specified in the debugging profile:
**CORBA**
CORBA object
**EJB** Enterprise bean
**Java** Java program
**LE** Compiled language program

**Netname**
On the "List LE profiles" page only, displays the contents of the **Netname** field.

**Class** On the "List Java profiles" and "List CORBA profiles" pages only, displays the contents of the **Class** field.

If the Class name is too long to display in the available space, the trailing characters are displayed, preceded by "**...**". To display the Class name in full, click the profile name.

**Bean** On the "List EJB profiles" page only, displays the contents of the **Bean** field.

If the bean name is too long to display in the available space, the leading characters are displayed, followed by "**...**." To display the bean name in full, click the profile name.

**Method**
On the "List EJB profiles" and "List CORBA profiles" pages only, displays the contents of the **Method** field.

If the Method name is too long to display in the available space, the leading characters are displayed, followed by "**...**". To display the Method name in full, click the profile name.

You can change the way information is displayed on the "List profiles" page:

**Selecting which profiles are displayed**

Use the checkboxes at the top of the page to select which debugging profiles are displayed. The options are:

- Display all profiles
- Display all profiles that you created
- Display all active profiles
- Display only active profiles that you created

**Sorting the list**

Use the buttons above each column to re-display the list in the sequence determined by the contents of the column. For example, to re-display the profiles in sequence of program name, click the **Program** button. CICS uses the EBCDIC sorting sequence when it re-displays the list.

Your choice of which profiles are displayed, and your chosen sequence, are preserved, and used the next time you use the debugging profile manager.

**Related tasks**:

"Creating a debugging profile with the Web interface" on page 24

"Activating debugging profiles with the Web interface" on page 25

"Inactivating debugging profiles with the Web interface" on page 26

"Viewing a debugging profile with the Web interface" on page 26

"Changing a debugging profile with the Web interface" on page 27

"Copying debugging profiles with the Web interface" on page 27

"Deleting debugging profiles with the Web interface" on page 28

"Setting the display device" on page 36

When you have created a debugging profile for a compiled language (Language Environment) program, but before you can start debugging the application programs defined in the profile, you must specify the display device with which you will interact with the debugger.

"Combining actions on the List debugging profiles screen" on page 46

**Related reference**:

"The Create compiled profile page" on page 29

"The Create Java profile page" on page 32

"The View LE profile page" on page 35

"The View Java profile page" on page 36

"The Set compiled display device page" on page 38

**Buttons on the List profiles page:** The buttons at the head of the following columns are used to re-display the list of profiles in sequence:

**Owner**
**Profile**
**Status**
**Tranid**
**Program**
**Compile Unit**
**Applid**
**Userid**
**Termid**
**Netname**
**Type**

The following buttons are inactive, and cannot be used to re-sequence the list of profiles:

**Class**

**Bean**

**Method**

Other actions are performed using the buttons at the bottom of the "List profiles" page:

**Activate**
Activate selected profiles. See "Activating debugging profiles with the Web interface" on page 25 for more information.

**Inactivate**
Inactivate selected profiles. See "Inactivating debugging profiles with the Web interface" on page 26 for more information.

**Copy**
Copy selected profiles. See "Copying debugging profiles with the Web interface" on page 27 for more information.

**Delete**
Delete selected profiles. See "Deleting debugging profiles with the Web interface" on page 28 for more information.

**Select all**
Selects all the profiles in the list.

**Deselect all**
Deselects all the profiles in the list

**Refresh**
Refresh the "List profiles" page. The list is updated to show any changes that you, and other users, have made.

## Creating a debugging profile with the Web interface
### About this task

You can create debugging profiles in these ways:
- You can create a completely new profile by entering all the information needed to define the profile
- You can base the new profile on an existing profile

**Related reference**:

"The List profiles page" on page 20

**Creating a new profile:**
**About this task**

Starting with the "List all profiles" page, follow these steps:

**Procedure**

- To create a profile for a compiled language (Language Environment®) program
  1. Click **Create compiled profile**. CICS displays the "Create compiled profile" page.
  2. Complete the information that you need to specify your profile. See "The Create compiled profile page" on page 29 for details.
  3. Click the **Create** button. CICS checks that you have entered valid data.
     – If your data is valid, the profile is saved

- If your data contains an error, a message is displayed. Re-enter the data, and click the **Create** button again.

4. Click **List all profiles** to return to the "List all profiles" page.

- To create a profile for a Java program
  1. Click **Create Java profile**. CICS displays the "Create Java profile" page.
  2. Complete the fields that you need to specify your profile.
  3. Click the **Create** button. CICS checks that you have entered valid data.
     - If your data is valid, the profile is saved
     - If your data contains an error, a message is displayed. Re-enter the data, and click the **Create** button again.
  4. Click **List all profiles** to return to the "List profiles" page.

**Basing a new profile on an existing profile:**
**About this task**

You can create a new debugging profile using an existing profile as a starting point. The steps you take depend upon whether the original profile is owned by you or another user:

**If you own the profile**
Follow the steps described in "Changing a debugging profile with the Web interface" on page 27. Before you save the profile, give it a new name.

**If another user owns the profile:**
Follow the steps below.

**Procedure**

1. Copy the profile, following the steps described in "Copying debugging profiles with the Web interface" on page 27.
2. Make any changes to the profile by following the steps described in "Changing a debugging profile with the Web interface" on page 27.

## Activating debugging profiles with the Web interface
### About this task

To activate debugging profiles, start with the "List all profiles" page, and follow these steps:

### Procedure

1. Use the check boxes at the top of the page to ensure that the display includes the profiles you want to activate.
2. Scroll the list to a profile that you want to activate.
3. Select the profile using the check box to the left of the profile name.
4. Repeat steps 2 through 3 to select all the profiles you want to activate.
5. Click the **Activate** button. By default, if any of the selected profiles is for a compiled language (Language Environment) program, CICS displays the "Set compiled display device" page.

   If none of the selected profiles is for a compiled language programs, CICS refreshes the "List all profiles" page.

   You can choose not to see the "Set compiled display device" page when you activate profiles. See "Setting the display device" on page 36 for more information.

### Results

If you change a profile while it is active, the changes take effect immediately: the next time a program is started, the changed parameters are used to decide if the program should run under the debugger's control.

**Related tasks**:

"Inactivating debugging profiles with the 3270 interface" on page 44

"Setting the display device" on page 36
When you have created a debugging profile for a compiled language (Language Environment) program, but before you can start debugging the application programs defined in the profile, you must specify the display device with which you will interact with the debugger.

**Related reference**:

"The List profiles page" on page 20

## Inactivating debugging profiles with the Web interface

### About this task

To inactivate debugging profiles, start with the "List profiles" page, and follow these steps:

### Procedure

1. Use the check boxes at the top of the page to ensure that the display includes the profiles you want to inactivate.
2. Scroll the list to a profile that you want to inactivate.
3. Select the profile using the check box to the left of the profile name.
4. Repeat steps 2 through 3 to select all the profiles you want to inactivate.
5. Click the **Inactivate** button.

### Results

The "List profiles" page is refreshed.

**Related reference**:

"The List profiles page" on page 20

## Viewing a debugging profile with the Web interface

### About this task

If you are not the owner of a debugging profile, you can view its contents, but you cannot change it. To view the contents of a debugging profile, start with the "List all profiles" page, and follow these steps:

### Procedure

1. Use the check boxes at the top of the page to ensure that the profile you want to view is displayed.
2. Scroll the list to the profile you want to view.
3. Click the profile name. CICS displays the "View compiled profile" page, or the "View Java profile" page.
4. Click **List all profiles** to return to the "List all profiles" page.

### Results

If you follow these steps for a profile that you own, CICS displays the "Edit compiled profile" page, or the "Edit Java profile" page. In this case, you will be able to modify the contents of the profile.

**Related reference**:

"The List profiles page" on page 20

## Changing a debugging profile with the Web interface

### About this task

If you are the owner of a debugging profile, you can change its contents. Starting with the "List all profiles" page, follow these steps:

### Procedure

1. Use the check boxes at the top of the page to ensure that the profile you want to change is displayed.
2. Scroll the list to the profile you want to change.
3. Click the profile name. CICS displays the "Create compiled profile" page, or the "Create Java profile" page.
4. Make your changes to the displayed fields.
5. Click the **Replace** button.
6. Click **List all profiles** to return to the "List all profiles" page.

### Results

**Note:**

1. If you change the name of the profile, the debugging profile manager creates a new profile with the new name, and leaves the original profile unchanged.
2. If you follow these steps for a profile that you do not own, CICS displays the "View compiled profile" page, or the "View Java profile" page. In this case, you will not be able to modify the contents of the profile.
3. If you change a profile while it is active, the changes take effect immediately: the next time a program is started, the changed parameters are used to decide if the program should run under the debugger's control.

**Related reference**:

"The List profiles page" on page 20

## Copying debugging profiles with the Web interface

### About this task

You can copy profiles that are owned by other users to create identical profiles that you own. You cannot copy a profile that you own. Each new profile has the same name as the one that was copied. Starting with the "List profiles" page, follow these steps:

### Procedure

1. Use the check boxes at the top of the page to ensure that the display includes the profiles you want to copy.
2. Scroll the list to a profile that you want to copy.
3. Select the profile using the check box to the left of the profile name.
4. Repeat steps 2 through 3 to select all the profiles you want to copy.

5. Click the **Copy** button.

### Results

The profiles are copied and the "List profiles" page is refreshed.

### What to do next

If you want to create a new profile based on a profile that you own, follow the steps described in "Changing a debugging profile with the Web interface" on page 27.

**Related reference**:

"The List profiles page" on page 20

## Deleting debugging profiles with the Web interface

### About this task

To delete debugging profiles, start with the "List profiles" page, and follow the steps below. You cannot delete a profile that is owned by another user.

### Procedure

1. Use the check boxes at the top of the page to ensure that the display includes the profiles you want to delete.
2. Scroll the list to a profile you want to delete.
3. Select the profile using the check box to the left of the profile name.
4. Repeat steps 2 through 3 to select all the profiles you want to delete.
5. Click the **Delete** button.

### Results

The profiles are deleted, and the "List profiles" page is refreshed.

**Related reference**:

"The List profiles page" on page 20

#### Deleting the sample profiles:

Although, in general, you cannot delete debugging profiles that are owned by another user, the sample profiles are handled as a special case, and you can delete them.

#### About this task

Be aware that, if you do delete the sample profiles:

- You may affect users who want to use the sample profiles.
- The only way to create them again is to have your system programmer re-initialize the debugging profiles data sets. However, if you do this, any other profiles which already exist will be deleted.

If you want to use the sample profiles, and you are concerned that other users may delete them, copy the samples. You will own the copies, and no-one else will be able to delete them.

If you don't want to see the sample profiles when you list profiles, display only the profiles that you own.

## The Create compiled profile page

Use the "Create compiled profile" page to work with the contents of a debugging profile for compiled language (Language Environment) programs. You can use the page to perform the following functions:

**Create a new profile**
Initially, the page contains default values for some of the fields. You must supply other values to complete the profile.

**Edit an existing profile**
Initially, the page contains the values that were previously defined for the profile.

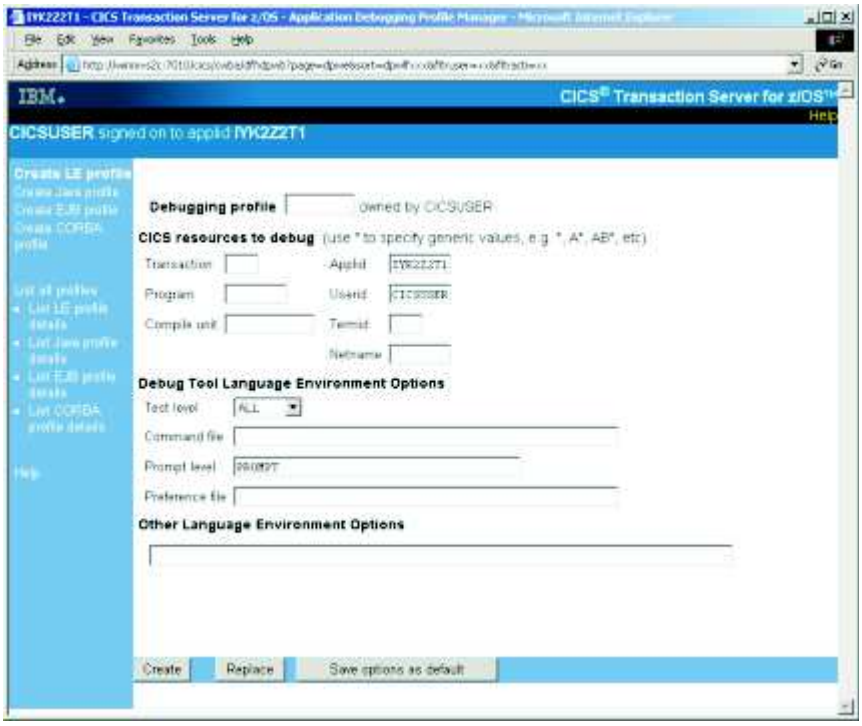



*Figure 2. The Create compiled profile page*

**Related reference**:

"The List profiles page" on page 20

**Fields on the Create compiled profile page:**
The fields on the "Create compiled profile" page are:

**`Debugging profile`**
Specifies the name of the profile.

If you are working with an existing profile, and you change the name that is displayed, the application debugging profile manager creates a new profile with the new name, and leaves the original profile unchanged.

| **Acceptable characters** |
| --- |
| `A-Z 0-9 $ @ #` |
| Any lowercase characters you enter are converted to uppercase. |

The following fields specify which programs should trigger the start of a debugging session when the profile is active.

**Transaction**
Specify a value in this field when you want to debug only those programs that run under the specified transaction id.

| **Acceptable characters** |
| --- |
| A-Z a-z 0-9 $ @ # . / - _ % & ¢ ? ! : \| " = ¬ , ; < > |
| Because transaction ids are case-sensitive, any lowercase characters you enter are **not** converted to uppercase. |

You can specify a generic value if you want to debug programs that run under a set of similarly-named transactions.

**Note:** Do not specify an alias transaction name in this field; CICS does not support the use of alias transaction names to select programs for debugging.

**Program**
Specify a value in this field when you want to debug only the specified program. In this context the program is the program as known to CICS, such as a load module name, initial program in a transaction or a program that has been XCTL'd or LINKed to.

| **Acceptable characters** |
| --- |
| A-Z 0-9 $ @ # |
| Any lowercase characters you enter are converted to uppercase. |

You can specify a generic value if you want to debug a set of similarly-named programs.

**Compile unit**
Specify a value in this field when you want to debug only the specified compile unit. You can specify a generic value if you want to debug a set of similarly-named compile units. In this context the compile unit is the program as known to the compiler; for example, PROGRAM-ID for COBOL and the main PROCEDURE name for PL/I.

| **Acceptable characters** |
| --- |
| A-Z a-z 0-9 $ @ # . / - _ % & ¢ ? ! : \| " = ¬ , ; < > |
| Because compile unit names are case-sensitive, any lowercase characters you enter are **not** converted to uppercase. |

**Applid**
Specify a value in this field when you want to confine debugging to programs that run in the specified CICS region.

| **Acceptable characters** |
| --- |
| A-Z 0-9 $ @ # |
| Any lowercase characters you enter are converted to uppercase. |

You can specify a generic value if you want to debug programs in a set of similarly-named regions.

The default value is the applid of the region where the application debugging profile manager is running, and is displayed at the top of the page.

**Userid**
Specify a value in this field when you want to confine debugging to programs that are being run by the specified user. The default value is the ID of the user that is using the debugging profile manager.

| **Acceptable characters** |
| --- |
| A-Z 0-9 $ @ # |
| Any lowercase characters you enter are converted to uppercase. |

You can specify a generic value if you want to debug programs that are being run by a group of similarly-named users.

The default value is the ID of the user that is using the application debugging profile manager, and is displayed at the top of the page.

**Important:** The user ID specified here is not necessarily the owner of the profile: the owner of the profile is the user who created it.

**Termid**
Specify a value in this field when you want to confine debugging to programs that are being run at the specified terminal.

| **Acceptable characters** |
| --- |
| A-Z a-z 0-9 $ @ # . / - _ % & ¢ ? ! : \| " = ¬ , ; < > |
| **Restriction:** You cannot specify a terminal ID that consists entirely of blanks |

You can specify a generic value if you want to debug programs that are being run at a number of similarly-named terminals.

**Important:** The terminal specified here is not necessarily the terminal at which the debugging session is conducted. The terminal used for the debugging session is specified in the "Set compiled display device" page.

**Netname**
Specify a value in this field when you want to confine debugging to programs that are being run at terminals with the specified netname.

| **Acceptable characters** |
| --- |
| A-Z 0-9 $ @ # |
| Any lowercase characters you enter are converted to uppercase. |

You can specify a generic value if you want to debug programs that are being run at a number of terminals with similar netnames.

The following fields specify options that are passed to Debug Tool. See *Debug Tool for z/OS and OS/390 User's Guide* for more information. You can save the values that you specify; the saved values are used by default each time you create a Language Environment debugging profile.

**Test level**
Specifies what conditions need to be met for Debug Tool to gain control for programs that match this profile. Select one of the following values:
**ALL**
**ERROR**
**NONE**

**Command file**
Specifies the primary commands file associated with the profile. You can specify the fully qualified name of a sequential data set or a member of a partitioned data set.

**Prompt level**
Specifies whether an initial commands list is unconditionally executed during program initialization. Enter one of the following:
**PROMPT**
**NOPROMPT**
*command*

**Preference file**
Specifies the preference file that Debug Tool uses when debugging programs that match this profile. You can specify the fully qualified name of a sequential data set or a member of a partitioned data set.

**Language Environment options**
Specifies Language Environment runtime options for programs that match this profile. When a program is selected for debugging because it matches the profile, the runtime options specified will override other runtime options that you may have in effect. For more information about defining runtime options for Language Environment, see Defining runtime options for Language Environment .

**Buttons on the Create compiled profile page:**
The buttons on the "Create compiled profile" page are:

**Create**
Create a new profile using the information entered on the page

**Replace**
Update an existing profile using the information entered on the page

**Save options as default**
Save the contents of the following fields. The saved values are used by default each time you create a Language Environment debugging profile.
**Test level**
**Command file**
**Prompt**
**Preference file**
**Language Environment options**

## The Create Java profile page

Use the "Create Java profile" page to work with the contents of a debugging profile for Java programs. You can use the page to perform the following functions:

**Create a new profile**
Initially, the page contains default values for some of the fields. You must supply other values to complete the profile.

**Edit an existing profile**
Initially, the page contains the values that were previously defined for the profile.

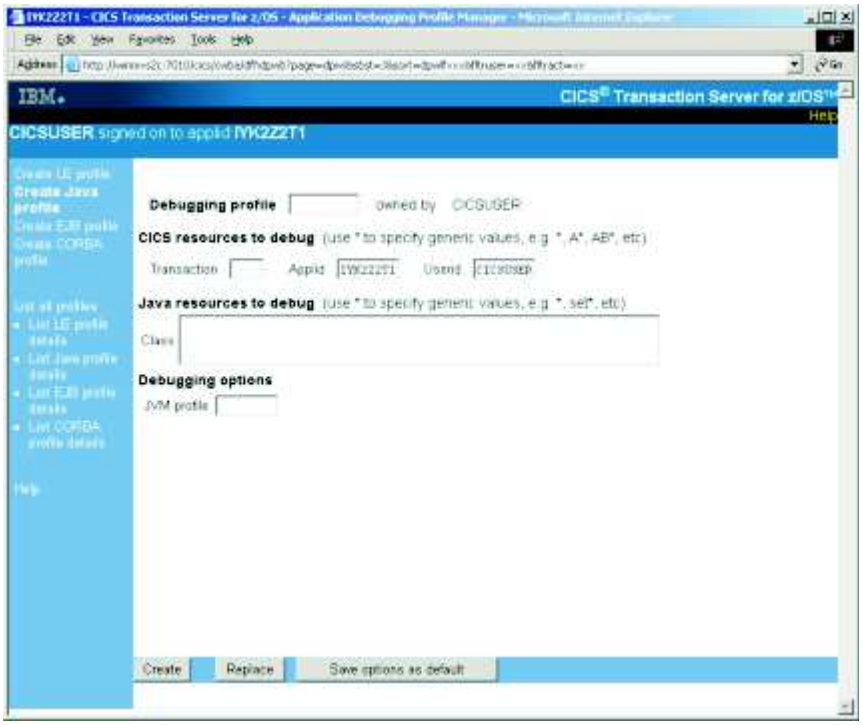



*Figure 3. The Create Java profile page*

**Related reference**:

"The List profiles page" on page 20

**Fields on the Create Java profile page:**
The fields on the "Create Java profile" page are:

**`Debugging Profile`**
Specifies the name of the profile.

If you are working with an existing profile, and you change the name that is displayed, the application debugging profile manager creates a new profile with the new name, and leaves the original profile unchanged.

| **Acceptable characters** |
| --- |
| `A-Z 0-9 $ @ #`<br><br>Any lower case characters you enter are converted to upper case. |

The following fields specify which programs should trigger the start of a debugging session when the profile is active:

**`Transaction`**
Specify a value in this field when you want to debug only those programs that run under the specified transaction id.

| **Acceptable characters** |
| --- |
| `A-Z a-z 0-9 $ @ # . / - _ % & ¢ ? ! : \| " = ¬ , ; < >` |

You can specify a generic value if you want to debug programs that run under a set of similarly name transactions.

**Note:** Do not specify an alias transaction name in this field; CICS does not support the use of alias transaction names to select programs for debugging.

**Applid**
Specify a value in this field when you want to confine debugging to programs that run in the specified CICS region. The default value is the applid of the region where the Debugging Profile Manager is running.

| **Acceptable characters** |
| --- |
| A-Z 0-9 $ @ # |
| Any lower case characters you enter are converted to upper case. |

You can specify a generic value if you want to debug programs in a set of similarly-named regions.

**Userid**
Specify a value in this field when you want to confine debugging to programs that are being run by the specified user. The default value is the ID of the user that is using the debugging profile manager.

| **Acceptable characters** |
| --- |
| A-Z 0-9 $ @ # |
| Any lower case characters you enter are converted to upper case. |

You can specify a generic value if you want to debug programs that are being run by a group of similarly-named users.

**Important:** The user ID specified here is not necessarily the owner of the profile: the owner of the profile is the user who created it.

The following fields specify which Java resources should trigger the start of a debugging session when the profile is active:

**Type**
Specifies the type of Java resource that you want to debug:
**Java** Select this value when you want to debug a Java program.
**EJB** Select this value when you want to debug an enterprise bean.
**CORBA**
Select this value when you want to debug a stateless CORBA object.

**Class**
For Java programs and stateless CORBA objects only, specify a value in this field when you want to debug only the specified class. You can specify a generic value if you want to debug a set of similarly-named classes.

**Bean**
For enterprise beans only, specify a value in this field when you want to debug only the specified bean. You can specify a generic value if you want to debug a set of similarly-named beans.

**Method**
For enterprise beans and stateless CORBA objects only, specify a value in this field when you want to debug only the specified method.

When an inbound request initiated by a Java remote method invocation is received, the value specified is compared with the mangled name in the inbound request to determine if the profile matches the request. If it is possible that mangling can take place, do not specify a method name in the debugging profile, but specify a generic method instead.

The following field specifies the debugging options for this profile. You can save the value that you specify; the saved value is used by default each time you create a Java debugging profile.

**JVM profile**
Specifies the name of the JVM profile that is used for Java programs that match this profile. The profile should specify that the Java program is to run in debug mode. You cannot specify a generic value for this parameter.

**Buttons on the Create Java profile page:**
The buttons on the "Create Java profile" page are:

**Create**
Create a new profile using the information entered on the page

**Replace**
Update an existing profile using the information entered on the page

**Save options as default**
Save the contents of the following field. The saved value is used by default each time you create a Java debugging profile.
**JVM profile**

## The View LE profile page

Use the "View LE profile" page to view the contents of a debugging profile for compiled language programs. You cannot change the profile with this page. The information displayed is described in "The Create compiled profile page" on page 29.

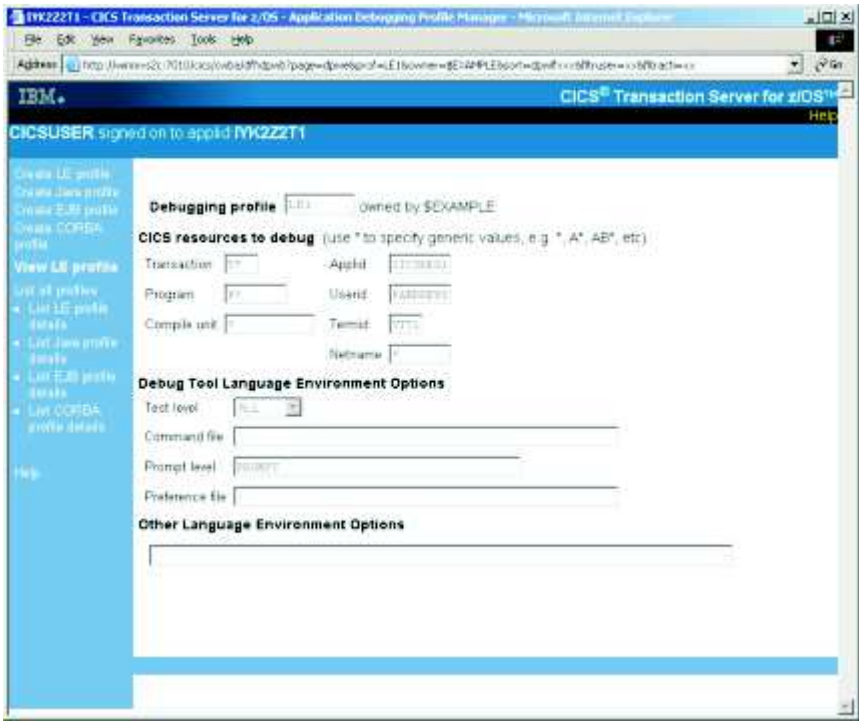



*Figure 4. The View LE profile page*

**Related reference**:
"The List profiles page" on page 20

## The View Java profile page

Use the "View Java profile" page to view the contents of a debugging profile for compiled language programs. You cannot change the profile with this page. The information displayed is described in "The Create Java profile page" on page 32.

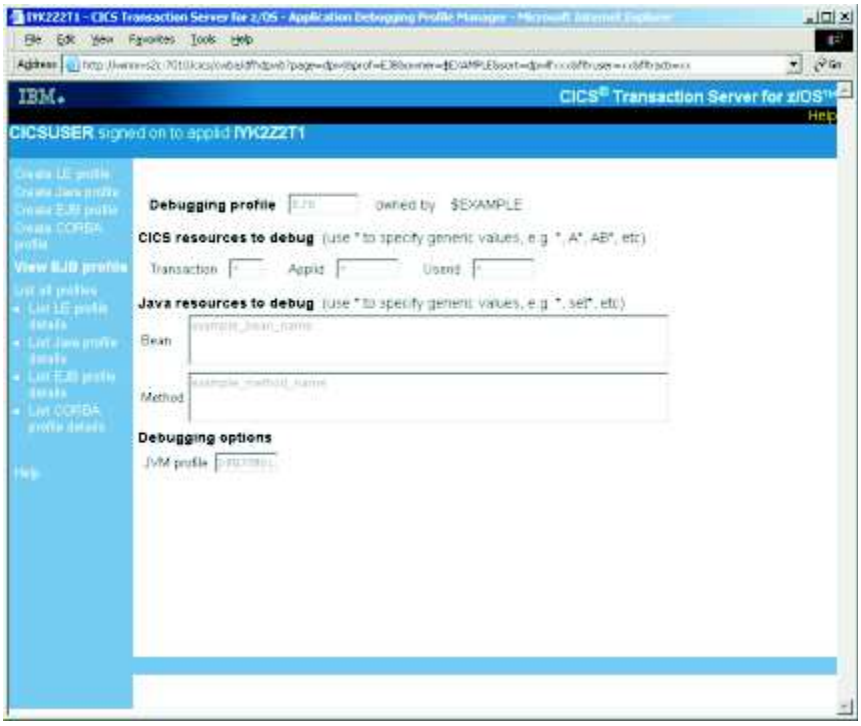



*Figure 5. The View Java profile page*

**Related reference**:

"The List profiles page" on page 20

## Setting the display device

When you have created a debugging profile for a compiled language (Language Environment) program, but before you can start debugging the application programs defined in the profile, you must specify the display device with which you will interact with the debugger.

### About this task

You can use one of the following as the display device:

- A 3270 terminal
- A debugging tool on a workstation

The display device becomes associated with a debugging profile when you activate the profile, and remains associated with the profile until you make the profile inactive.

You can choose when you specify the display device:

- You can specify the display device *before* you make a profile active. The same display device will be associated with each profile that you subsequently make active.

- You can specify the display device *when* you make a profile active. The same display device will be associated with the profile that is made active. If you make more than one profile active at the same time (by selecting a number of profiles on the "List debugging profile" page) the same display device will be associated with them all.

**Related tasks**:

"Activating debugging profiles with the Web interface" on page 25

**Related reference**:

"The List profiles page" on page 20

**Specifying the display device before you make a profile active:** Starting with the "List profiles" page, follow these steps:

1. Click Set LE display device. CICS displays the "Set LE display device" page.
2. Complete the details of the display device that you want to associate with the profile.
3. Select "In the future, do not show this page when activating profiles".
4. Click the **Save and return** button. CICS saves the display device settings, and displays the "List profiles" page.

The "Set LE display device" page will not be displayed when you activate profiles; the settings you have supplied will be applied to all profiles for compiled language programs when you activate them.

**Specifying the display device when you make a profile active:**
**About this task**

Starting with the "List profiles" page, follow these steps:

**Procedure**

1. Click Set LE display device. CICS displays the "Set LE display device" page.
2. Complete the details of the display device that you want to associate with the profile.
3. Using the check box, deselect "In the future, do not show this page when activating profiles".
4. Click the **Save and return** button. CICS saves the display device settings, and displays the "List profiles" page.

**Results**

CICS will display the "Set LE display device" page whenever you activate profiles for compiled language profiles. The page is initialized with the last settings you supplied.

**What to do next**

If you need to change the settings, follow these steps:

1. Change any details of the display device that you want to associate with the set of profiles that is being activated.
2. Click the **Save and return** button. CICS saves the display device settings, and displays the "List profiles" page.

## The Set compiled display device page

Use the "Set compiled display device" page to specify the display device with which you will interact with the debugger.

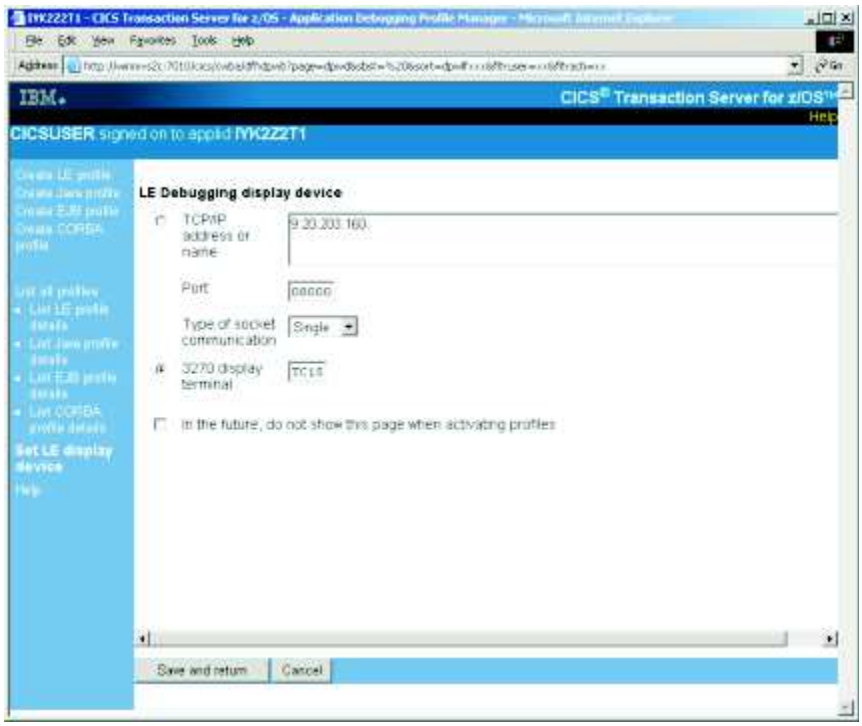



*Figure 6. The Set LE display device page*

**Related reference**:

"The List profiles page" on page 20

**Fields on the Set compiled display device page:**
The fields on the "Set compiled display device" page are:

**Debugging display device**
Use the radio buttons to select how you will interact with the debugger:

**TCP/IP address or name**
Specifies that you will interact with the debugger using a debugging client on your workstation. Supply the following information:

- The IP address or name of the host where the debugging client is running. By default, CICS inserts the IP address of the client which is running the browser, or — if there is a firewall between the browser and CICS — the IP address of the firewall.
- The port number at which the debugging client listens for a connection. Specify a value in the range 0 - 65535. The default is 8000.

**Type of socket communication**
For a debugging client on your workstation, specifies whether the debugging client and debugging server will communicate using a single socket or more than one socket.

**Single**
Use a single socket for communication. This is the default value, and is the preferred value when you use a WebSphere® Studio product as your debugging client.

**Multiple**
Use more than one socket for communication. You must specify this value when you use a VisualAge® product as your debugging client.

**3270 display terminal**
Specifies that you will interact with the debugger using a 3270 type terminal. Supply the following information:

- The terminal id of the terminal at which you will interact with the debugger.

  **Important:** The terminal specified here is not necessarily the terminal at which the transaction being debugged will run.

**Buttons on the Set compiled display device page:** The buttons on the "Set compiled display device" page are:

**Save and return**
Save the settings and return to the "List all profiles" page

**Cancel**
Return to the "List all profiles" page without saving the settings

# Using the application debugging profile manager 3270 interface

The CADP transaction displays debugging profile information by means of the application debugging profile manager 3270 interface.

## About this task

If you are debugging TCP/IP information, note that the port number must be specified in the PORT field of your resource. If the port is specified as part of the HOST attribute, the port information is not available to the CADP transaction.

## Procedure

1. On a 3270 screen, enter `CADP`.
2. Press `ENTER`. CICS displays the "List debugging profiles" screen.
3. Select your next action from the list of subtopics.

## The List debugging profiles screen

Use the "List debugging profiles" screen to display a list of debugging profiles. When you start the debugging profile manager, CICS displays the "List debugging profiles" screen.

```
CADP      -    CICS Application Debugging Profile Manager   -     IYK2Z2T1

List Debugging Profiles    (A=Activate,I=Inactivate,D=Delete,C=Copy)

  Owner    Profile  S Tran Program  Compile Unit Applid   Userid   Term Type
_ $EXAMPLE CORBA    I T*                         *        IORWERTH      Corb
_ CICSUSER CORBA    I T*                         *        IORWERTH      Corb
_ $EXAMPLE EJB      I *                          *        *             EJB
_ $EXAMPLE JAVA     I TR*                        *        PENFOLD*      Java
_ $EXAMPLE LE1      I T*   P*       *            CICSREG1 PANDREWS TTT1 LE
_ $EXAMPLE LE2      I TR   *        SAMPCOMPUN + CICSREG2 DRBEARD* TTT2 LE
_ $EXAMPLE LE3      I TRN3 PROG3    *            CICSREG3 *        TTT2 LE









7   profile(s). All profiles shown
Enter=Process PF1=Help 2=Filter 3=Exit 4=View 5=Create LE 6=Create Java
                9=Set display device 10=Edit 11=Sort
```

*Figure 7. The List debugging profiles screen, showing the example profiles*

When you use the debugging profile manager for the first time, CICS displays all the profiles that you own. Subsequently, CICS displays the profiles that were selected when you last used it.

If there are more profiles than can be displayed on one screen, use PF7 and PF8 to scroll backwards and forwards through the list. If you have no profiles, CICS displays an empty list.

The list contains selected information from the debugging profiles. The columns on the screen are:

**Owner**
The userid of the profile owner; that is, of the user who created the profile.

**Profile**
The name of the profile

**S** The status of the profile (**A** for Active, or **I** for Inactive)

The following columns display information specified when the profile is created:

**Tran** Displays the contents of the **transaction** field

**Program**
Displays the contents of the **program** field

**Compile Unit**
Displays the the first ten characters of the **Compile Unit** field. If the **Compile Unit** name is longer, a + to the right of the name shows that only part of the name is displayed.

**Applid**
Displays the contents of the **Applid** field

**Userid**
Displays the contents of the **Userid** field

**Term** Displays the contents of the **Terminal** field

**Type** Displays the type of program specified in the debugging profile:

**Corb** CORBA object
**EJB** Enterprise bean
**Java** Java program
**LE** Compiled language program

Not all the information in the debugging profile is displayed on the "List debugging profiles" screen. To display the additional information, move the cursor to the line that contains the profile, and press PF4.

You can change the way CICS displays information on the "List debugging profiles" screen:

**Selecting which profiles are displayed**

Use PF2 to cycle through the available options in turn. The options are

1. Display all the profiles in the system. This is the setting the first time you use the debugging profile manager
2. Display the profiles that you own
3. Display all active profiles

**Sorting the list**

Use PF11 to cycle through the available options in turn. The options are:

1. Re-display the profiles in sequence of profile name. This is the sequence the first time you use the debugging profile manager
2. Re-display the profiles in sequence of transaction ID
3. Re-display the profiles in sequence of program name
4. Re-display the profiles in sequence of owner

In each case, CICS uses the EBCDIC sorting sequence.

Your choice of which profiles are displayed, and your chosen sequence, are preserved, and used the next time you use the debugging profile manager.

**Related tasks**:

"Creating a debugging profile with the 3270 interface" on page 42
"Activating debugging profiles with the 3270 interface" on page 43
"Inactivating debugging profiles with the 3270 interface" on page 44
"Viewing a debugging profile with the 3270 interface" on page 44
"Changing a debugging profile with the 3270 interface" on page 44
"Copying debugging profiles with the 3270 interface" on page 45
"Deleting debugging profiles with the 3270 interface" on page 45
"Setting the display device" on page 56

**Related reference**:

"The Create Java debugging profile screen" on page 51
"The Create LE debugging profile screen" on page 47
"The View Java debugging profile screen" on page 54
"The View LE debugging profile screen" on page 54
"The Set LE debugging display device screen" on page 57

**Function keys for the List debugging profiles screen:** The function keys for the "List debugging profiles" screen are:

**PF1**
Display the help screen

**PF2**
Selects which debugging profiles are displayed. This key cycles through the available options in turn.

**PF3**
End the debugging profile manager
**PF4**
Displays the "View LE debugging profile" screen or the "View Java debugging profile" screen for the profile on the line that contains the cursor.
**PF5**
Create a new debugging profile for a compiled language program
**PF6**
Create a new debugging profile for a Java program
**PF7**
Scroll backwards
**PF8**
Scroll forwards
**PF9**
Display the "Set LE debugging display device" screen
**PF10**
Edit the profile on the line that contains the cursor, using the "Create LE debugging profile" screen or the "Create Java debugging profile" screen.
**PF11**
Re-display the debugging profiles in a different sequence. This key cycles through the available options in turn.

## Creating a debugging profile with the 3270 interface

### About this task

You can create debugging profiles in these ways:

### Procedure

- You can create a completely new profile by entering all the information needed to define the profile.
- You can base the new profile on an existing profile.

**Related reference**:

"The List debugging profiles screen" on page 39

"The Create LE debugging profile screen" on page 47

"The Create Java debugging profile screen" on page 51

**Creating a new profile:**
**About this task**

Starting with the "List debugging profiles" screen, follow these steps:

**Procedure**

1. Decide which type of profile you want to create.
   - To create a profile for a compiled language program, press PF5. CICS displays the "Create LE debugging profile" screen.
   - To create a profile for a Java program, press PF6. CICS displays the "Create Java debugging profile" screen.
2. Complete the fields that you need to specify your profile. You might need to use PF7 and PF8 to scroll the display.
3. Press ENTER. CICS checks that you have entered valid data.
   - If you have specified valid data, the profile is saved
   - If your data contains an error, CICS displays a message. Re-enter the data, and press ENTER again.

4. Press PF12 to return to the “List debugging profiles” screen.

**Basing a new profile on an existing profile:**
**About this task**

You can create a new debugging profile using an existing profile as a starting point. The steps you take depend upon whether the original profile is owned by you or another user:

**If you own the profile**
Follow the steps described in “Changing a debugging profile with the 3270 interface” on page 44. Before you save the profile, give it a new name.

**If another user owns the profile**
Follow the steps below.

**Procedure**

1. Copy the profile, following the steps described in “Copying debugging profiles with the 3270 interface” on page 45.
2. Make any changes to the profile by following the steps described in “Changing a debugging profile with the 3270 interface” on page 44.

## Activating debugging profiles with the 3270 interface
### About this task

To activate debugging profiles, start with the “List debugging profiles” screen, and follow these steps:

### Procedure

1. Use PF2 to ensure that the display includes the profiles you want to activate.
2. Use PF7 and PF8 to scroll to a profile that you want to activate.
3. Type **A** (for Activate) in the field to the left of the profile name.
4. Repeat steps 2 through 3 to select all the profiles you want to activate.
5. Press ENTER. By default, if any of the selected profiles is for a compiled language program, CICS displays the “Set LE debugging display device” screen.

   If none of the selected profiles is for a compiled language program, CICS refreshes the “List debugging profiles” screen.

   You can choose not to see the “Set LE debugging display device” screen when you activate profiles. See “Setting the display device” on page 56 for more information.

### Results

**Note:** If you change a profile while it is active, the changes take effect immediately: the next time a program is started, the changed parameters are used to decide if the program should run under the debugger's control.

**Related reference**:

“The List debugging profiles screen” on page 39

“The Set LE debugging display device screen” on page 57

## Inactivating debugging profiles with the 3270 interface

### About this task

To inactivate debugging profiles, start with the "List debugging profiles" screen, and follow these steps:

### Procedure

1. Use PF2 to ensure that the display includes the profiles you want to inactivate.
2. Use PF7 and PF8 to scroll to a profile that you want to inactivate.
3. Type **I** (for Inactivate) in the field to the left of the profile name.
4. Repeat steps 2 through 3 to select all the profiles you want to inactivate.
5. Press ENTER. CICS makes the selected profiles inactive, and refreshes the "List debugging profiles" screen.

**Related tasks**:

"Activating debugging profiles with the Web interface" on page 25

**Related reference**:

"The List debugging profiles screen" on page 39

## Viewing a debugging profile with the 3270 interface

### About this task

To view the contents of a debugging profile, without changing it, start with the "List debugging profiles" screen, and follow these steps:

### Procedure

1. Use PF2 to ensure that the display includes the profile you want to view.
2. Use PF7 and PF8 to scroll to the profile you want to view.
3. Move the cursor to the line containing the profile that you want to view.
4. Press PF4. CICS displays the "View LE debugging profile" screen, or the "View Java debugging profile", depending on the options that were selected when the debugging profile was created.
5. When you have finished viewing the profile, press PF12 to return to the "List debugging profiles" screen.

**Related reference**:

"The List debugging profiles screen" on page 39

"The View Java debugging profile screen" on page 54

"The View LE debugging profile screen" on page 54

## Changing a debugging profile with the 3270 interface

### About this task

If you are the owner of a debugging profile, you can change its contents. Starting with the "List debugging profiles" screen, follow these steps:

### Procedure

1. Use PF2 to ensure that the display includes the profile you want to change.
2. Use PF7 and PF8 to scroll to a profile you want to change.
3. Move the cursor to the line containing the profile that you want to change.

4. Press PF10. CICS displays “Create LE debugging profile” screen, or the “Create Java debugging profile” screen, depending on the options that were selected when the debugging profile was created.
5. Make your changes to the fields displayed on the screen.
6. Press PF10. CICS saves the changed profile.
7. Press PF12 to return to the “List debugging profiles” screen.

### Results

**Note:**

1. If you change the name of the profile, the debugging profile manager creates a new profile with the new name, and leaves the original profile unchanged.
2. If you change a profile while it is active, the changes take effect immediately: the next time a program is started, the changed parameters are used to decide if the program should run under the debugger's control.

**Related reference**:

“The List debugging profiles screen” on page 39

“The Create LE debugging profile screen” on page 47

“The Create Java debugging profile screen” on page 51

## Copying debugging profiles with the 3270 interface

### About this task

You can copy profiles that are owned by other users, to create identical profiles that you own. Each new profile has the same name as the one that was copied. Starting with the “List debugging profiles” screen, follow these steps:

### Procedure

1. Use PF2 to ensure that the display includes the profiles you want to copy.
2. Use PF7 and PF8 to scroll to a profile that you want to copy.
3. Type **C** (for Copy) in the field to the left of the profile name.
4. Repeat steps 2 through 3 to select all the profiles you want to copy.
5. Press ENTER. CICS copies the profiles, and refreshes the “List debugging profiles” screen.

### Results

**Note:** You cannot copy a profile that you own. If you want to create a new profile based on a profile that you own, follow the steps described in “Changing a debugging profile with the 3270 interface” on page 44. Before saving the changed profile, give it a new name.

**Related tasks**:

“Combining actions on the List debugging profiles screen” on page 46

**Related reference**:

“The List debugging profiles screen” on page 39

## Deleting debugging profiles with the 3270 interface

### About this task

To delete debugging profiles, start with the “List debugging profiles” screen, and follow the steps below. You cannot delete a profile that is owned by another user.

### Procedure

1. Use PF2 to ensure that the display includes the profiles you want to delete.
2. Use PF7 and PF8 to scroll to a profile you want to delete.
3. You must make the profile inactive before you can delete it. To do this, see "Inactivating debugging profiles with the 3270 interface" on page 44.
4. Type **D** (for Delete) in the field to the left of the profile name.
5. Repeat steps 2 through 4 to select all the profiles you want to delete.
6. Press ENTER. CICS deletes the selected profiles, and refreshes the "List debugging profiles" screen.

**Related reference**:

"The List debugging profiles screen" on page 39

**Deleting the sample profiles:**
**About this task**

Although, in general, you cannot delete debugging profiles that are owned by another user, the sample profiles are handled as a special case, and you can delete them. Be aware that, if you do delete the sample profiles:

- You may affect users who want to use the sample profiles.
- The only way to create them again is to have your system programmer re-initialize the debugging profiles data sets. However, if you do this, any other profiles which already exist will be deleted.

If you want to use the sample profiles, and you are concerned that other users may delete them, copy the samples. You will own the copies, and no-one else will be able to delete them.

If you don't want to see the sample profiles when you list profiles, use PF2 on the "List debugging profiles" screen to display only the profiles that you own.

## Combining actions on the List debugging profiles screen
### About this task

From the "List debugging profiles screen", you can activate, inactivate, delete and copy debugging profiles by typing the appropriate action character (**A**, **I**, **D** and **C** respectively) in the field to the left of the profile name. You can combine these actions on the "List debugging profiles" screen. For example, you can activate some profiles, and inactivate others in a single operation:

### Procedure

1. Use PF2 to ensure that the display includes the profiles you want to work with.
2. Use PF7 and PF8 to scroll to a profile you want to work with.
3. Type the action character in the field to the left of the profile name.
4. Repeat steps 2 and 3 to select all the profiles you want to work with.
5. Press ENTER. CICS processes the selected profiles, and refreshes the "List debugging profiles" screen. If any of the selected actions fail, processing stops, and the unprocessed action characters remain on the screen.

**Related tasks**:

"Copying debugging profiles with the 3270 interface" on page 45

**Related reference**:

"The List profiles page" on page 20

## The Create LE debugging profile screen

Use the "Create LE debugging profile" screen to work with the contents of a debugging profile for compiled language programs. You can use the screen to perform the following functions:

**Create a new profile**
Initially, the screen contains default values for some of the fields. You must supply other values to complete the profile.

**Edit an existing profile**
Initially, the screen contains the values that were previously defined for the profile.

```
CADP      -    CICS Application Debugging Profile Manager   -     IYK2Z2T1

Create LE Debugging Profile ==>           for CICSUSER


CICS Resources To Debug (use * to specify generic values e.g. *, A*, AB*, etc.)
Transaction      ==>                                  Applid   ==> IYK2Z2T1
Program          ==>                                  Userid   ==> CICSUSER
Compile Unit     ==>                                  Termid   ==> TC15
                                                      Netname  ==> IYCWTC15

Debug Tool Language Environment Options
Test Level       ==> All                                      (All,Error,None)
Command File     ==>
Prompt Level     ==> PROMPT
Preference File  ==>

Other Language Environment Options
==>
==>
==>
==>

Enter=Create PF1=Help 2=Save options as defaults 3=Exit 10=Replace 12=Return
```

*Figure 8. The CADP Create LE debugging profile screen*

**Related tasks**:

"Creating a debugging profile with the 3270 interface" on page 42

"Changing a debugging profile with the 3270 interface" on page 44

**Related reference**:

"The List debugging profiles screen" on page 39

**Fields on the Create LE debugging profile screen:** The fields on the "Create LE debugging profile" screen are:

**Create LE Debugging Profile**
Specify the name of the profile. If you change the name of an existing profile, CADP creates a new profile with the new name, and leaves the original profile unchanged.

| **Acceptable characters** |
| --- |
| A-Z 0-9 $ @ # |
| Any lowercase characters you enter are converted to uppercase. |

The following fields specify which programs should trigger the start of a debugging session when the profile is active.

**Transaction**
Specify a value in this field when you want to debug only those programs that run under the specified transaction id.

| **Acceptable characters** |
| --- |
| A-Z a-z 0-9 $ @ # . / - _ % & ¢ ? ! : \| " = ¬ , ; < > |
| Because transaction ids are case-sensitive, any lowercase characters you enter are **not** converted to uppercase. |

You can specify a generic value if you want to debug programs that run under a set of similarly-named transactions.

**Note:** Do not specify an alias transaction name in this field; CICS does not support the use of alias transaction names to select programs for debugging.

**Program**
Specify a value in this field when you want to debug only the specified program. In this context the program is the program as known to CICS, such as a load module name, initial program in a transaction or a program that has been XCTL'd or LINKed to.

| **Acceptable characters** |
| --- |
| A-Z 0-9 $ @ # |
| Any lowercase characters you enter are converted to uppercase. |

You can specify a generic value if you want to debug a set of similarly-named programs.

**Compile unit**
Specify a value in this field when you want to debug only the specified compile unit. You can specify a generic value if you want to debug a set of similarly-named compile units. In this context the compile unit is the program as known to the compiler; for example, PROGRAM-ID for COBOL and the main PROCEDURE name for PL/I.

| **Acceptable characters** |
| --- |
| A-Z a-z 0-9 $ @ # . / - _ % & ¢ ? ! : \| " = ¬ , ; < > |
| Because compile unit names are case-sensitive, any lowercase characters you enter are **not** converted to uppercase. |

**Applid**
Specify a value in this field when you want to confine debugging to programs that run in the specified CICS region. The default value is the applid of the region where the Debugging Profile Manager is running.

| **Acceptable characters** |
| --- |
| A-Z 0-9 $ @ # |
| Any lowercase characters you enter are converted to uppercase. |

You can specify a generic value if you want to debug programs in a set of similarly-named regions.

**Userid**
Specify a value in this field when you want to confine debugging to programs that are being run by the specified user. The default value is the ID of the user who is using the debugging profile manager.

| **Acceptable characters** |
| --- |
| A-Z 0-9 $ @ # |
| Any lowercase characters you enter are converted to uppercase. |

You can specify a generic value if you want to debug programs that are being run by a group of similarly-named users.

**Important:** The user ID specified here is not necessarily the owner of the profile: the owner of the profile is the user who created it.

**Termid**
Specify a value in this field when you want to confine debugging to programs that are being run at the specified terminal.

| **Acceptable characters** |
| --- |
| A-Z a-z 0-9 $ @ # . / - _ % & ¢ ? ! : \| " = ¬ , ; < > |
| **Restriction:** You cannot specify a terminal ID that consists entirely of blanks |

You can specify a generic value if you want to debug programs that are being run at a number of similarly-named terminals.

**Important:** The terminal specified here is not necessarily the terminal at which the debugging session is conducted. The terminal used for the debugging session is specified in the “Set LE debugging display device” screen.

**Netname**
Specify a value in this field when you want to confine debugging to programs that are being run at terminals with the specified netname.

| **Acceptable characters** |
| --- |
| A-Z 0-9 $ @ # |
| Any lowercase characters you enter are converted to uppercase. |

You can specify a generic value if you want to debug programs that are being run at a number of terminals with similar netnames.

The following fields specify suboptions of the TEST runtime option, and are passed to Debug Tool. See the *Debug Tool for z/OS and OS/390 User's Guide* for more information. You can save the values that you specify; the saved values are used by default each time you create an LE debugging profile. For more information, see “Specifying default values for Debug Tool and LE options” on page 50.

**Test level**
Specifies which conditions raised by your program will cause Debug Tool to gain control. You can enter the following values:
**All**
**Error**
**None**

**Command file**
Specifies the primary commands file associated with the profile. You can specify the fully qualified name of a sequential data set or a member of a partitioned data set.

**Prompt level**
Specifies whether an initial commands list is unconditionally executed during program initialization. Enter one of the following:
**PROMPT**
**NOPROMPT**
*command*

**Preference file**
Specifies the preference file that Debug Tool uses when debugging programs that match this profile. You can specify the fully qualified name of a sequential data set or a member of a partitioned data set.

**Other Language Environment options**
Specifies Language Environment runtime options for programs that match this profile. When a program is selected for debugging because it matches the profile, the runtime options specified will override other runtime options that you may have in effect. For more information, see Defining runtime options for Language Environment .

**Function keys for the Create LE debugging profile screen:** The function keys for the "Create LE debugging profile" screen are:
**PF1**
Display the help screen
**PF2**
Save the contents of the Debug Tool options, and the Language Environment options. See "Specifying default values for Debug Tool and LE options."
**PF3**
End the debugging profile manager
**PF10**
Update an existing profile with the information on the screen
**PF12**
Return to the "List debugging profiles" screen

*Specifying default values for Debug Tool and LE options:* You can specify default values for the following Debug Tool options, and the Language Environment options. The saved values are used by default each time you create a debugging profile for a compiled language program. The Debug Tool options are:
**Test level**
**Command file**
**Prompt level**
**Preference file**

To save the default values, start with the "Create LE debugging profile" screen, and follow these steps:

1. Type the default values that you want to specify for the Debug Tool options and Language Environment options
2. Press PF2. CICS saves the values that you have specified.

The values that you save will be used by default each time you create a new profile.

## The Create Java debugging profile screen

Use the "Create Java debugging profile" screen to work with the contents of a debugging profile for Java programs. You can use the screen to perform the following functions:

**Create a new profile**
Initially, the screen contains default values for some of the fields. You must supply other values to complete the profile.

**Edit an existing profile**
Initially, the screen contains the values that were previously defined for the profile.

The screen is in two parts; use PF7 and PF8 to scroll between them.

```
CADP      -    CICS Application Debugging Profile Manager   -     IYK2Z2T1

Create Java Debugging Profile ==>            for CICSUSER

CICS Resources To Debug (use * to specify generic values e.g. *, A*, AB*, etc.)
Transaction      ==>                          Applid       ==> IYK2Z2T1
                                              Userid       ==> CICSUSER

Debugging Options
JVM Profile      ==>

Java Resources To Debug
Type             ==> J       (J=Java Applications, E=Enterprise Beans, C=Corba)

Class (Java Applications or Corba)
==>
==>
==>
==>

Press PF8 to set Bean and Method

Enter=Create PF1=Help 2=Save options as defaults 3=Exit 8=Forward
10=Replace 12=Return
```

```
CADP      -    CICS Application Debugging Profile Manager   -     IYK2Z2T1

Java Resources To Debug

Bean (Enterprise Beans only)
==>
==>
==>
==>

Method (Enterprise Beans or Corba)
==>
==>
==>
==>









Enter=Create PF1=Help 3=Exit 7=Back 10=Replace 12=Return
```

*Figure 9. The CADP Create Java debugging profile screens*

**Related tasks**:

"Creating a debugging profile with the 3270 interface" on page 42
"Changing a debugging profile with the 3270 interface" on page 44

**Related reference**:

"The List debugging profiles screen" on page 39

**Fields on the Create Java debugging profile screen:** The fields on the "Create Java debugging profile" screen are:

**Create Java Debugging Profile**
Specifies the name of the profile. If you change the name of an existing profile, CADP creates a new profile with the new name, and leaves the original profile unchanged.

| **Acceptable characters** |
| --- |
| A-Z 0-9 $ @ # |
| Any lowercase characters you enter are converted to uppercase. |

The following fields specify which programs should trigger the start of a debugging session when the profile is active.

**Transaction**
Specify a value in this field when you want to debug only those programs that run under the specified transaction id.

| **Acceptable characters** |
| --- |
| A-Z a-z 0-9 $ @ # . / - _ % & ¢ ? ! : \| " = ¬ , ; < > |

You can specify a generic value if you want to debug programs that run under a set of similarly-named transactions.

**Note:** Do not specify an alias transaction name in this field; CICS does not support the use of alias transaction names to select programs for debugging.

**Applid**
Specify a value in this field when you want to confine debugging to programs that run in the specified CICS region. The default value is the applid of the region where the Debugging Profile Manager is running.

| **Acceptable characters** |
| --- |
| A-Z 0-9 $ @ # |
| Any lowercase characters you enter are converted to uppercase. |

You can specify a generic value if you want to debug programs in a set of similarly-named regions.

**Userid**
Specify a value in this field when you want to confine debugging to programs that are being run by the specified user. The default value is the ID of the user that is using the debugging profile manager.

| **Acceptable characters** |
| --- |
| A-Z 0-9 $ @ # |
| Any lowercase characters you enter are converted to uppercase. |

You can specify a generic value if you want to debug programs that are being run by a group of similarly-named users.

**Important:** The user ID specified here is not necessarily the owner of the profile: the owner of the profile is the user who created it.

The following field specifies the debugging options for this profile. You can save the value that you specify; the saved value is used by default each time you create a Java debugging profile. To save the value, press PF2.

**JVM profile**
Specifies the name of the JVM profile that is used for Java programs that match this profile. The profile should specify that the Java program is to run in debug mode.

The following fields specify which Java resources should trigger the start of a debugging session when the profile is active:

**Type**
Specifies the type of Java resource that you want to debug:
**J** Enter this value when you want to debug a Java program.
**E** Enter this value when you want to debug an enterprise bean.
**C** Enter this value when you want to debug a stateless CORBA object.

**Class**
For Java programs and stateless CORBA objects only, specify a value in this field when you want to debug only the specified class. You can specify a generic value if you want to debug a set of similarly-named classes.

**Bean**
For enterprise beans only, specify a value in this field when you want to debug only the specified bean. You can specify a generic value if you want to debug a set of similarly-named beans.

**Method**
For enterprise beans and CORBA objects only, specify a value in this field when you want to debug only the specified method.

When an inbound request initiated by a Java remote method invocation is received, the value specified is compared with the mangled name in the inbound request to determine if the profile matches the request. If it is possible that mangling can take place, do not specify a method name in the debugging profile, but specify a generic method instead.

**Function keys for the Create Java debugging profile screen:** The function keys for the "Create Java debugging profile" screen are:
**PF1**
Display the help screen
**PF2**
Save the contents of the **JVM profile** field. The saved value is used by default each time you create a Java debugging profile.
**PF3**
End the debugging profile manager
**PF7**
Scroll forward
**PF8**
Scroll backward
**PF10**
Update an existing profile with the information on the screen

**PF12**
Return to the "List debugging profiles" screen

## The View LE debugging profile screen

Use the "View LE debugging profiles" screen to view the contents of a debugging profile for compiled language programs. You cannot change the profile with this screen.

```
CADP      -    CICS Application Debugging Profile Manager   -     IYK2Z2T1

View LE Debugging Profile LE1      for $EXAMPLE


CICS Resources To Debug
Transaction      ==> T*                              Applid   ==> CICSREG1
Program          ==> P*                              Userid   ==> PANDREWS
Comp Unit        ==> *                               Termid   ==> TTT1
                                                     Netname  ==> *

Debug Tool Language Environment Options
Test Level       ==> All                                    (All,Error,None)
Command File     ==>
Prompt Level     ==> PROMPT
Preference File  ==>

Other Language Environment Options
==>
==>
==>
==>

PF1=Help 3=Exit 12=Return
```

*Figure 10. The View LE debugging profile screen, showing example profile LE1*

The information displayed is described in "The Create LE debugging profile screen" on page 47.

**Related tasks**:

"Viewing a debugging profile with the 3270 interface" on page 44

**Related reference**:

"The List debugging profiles screen" on page 39

**Function keys for the View LE debugging profile screen:** The function keys for the "View LE debugging profile" screen are:

**PF1**
Display the help screen

**PF3**
End the debugging profile manager

**PF12**
Return to the "List debugging profiles" screen

## The View Java debugging profile screen

Use the "View Java debugging profiles" screen to view the contents of a debugging profile for Java programs. You cannot change the profile with this screen.

The screen is in two parts; use PF7 and PF8 to scroll between them.

```
CADP      -    CICS Application Debugging Profile Manager   -     IYK2Z2T1

View Java Debugging Profile EJB      for $EXAMPLE

CICS Resources To Debug
Transaction      ==> *                        Applid       ==> *
                                              Userid       ==> *

Debugging Options
JVM Profile      ==> DFHJVM01

Java Resources To Debug
Type             ==> E       (J=Java Applications, E=Enterprise Beans, C=Corba)

Class (Java Applications or Corba)
==>
==>
==>
==>




PF1=Help 3=Exit 8=Forward 12=Return
```

```
CADP      -    CICS Application Debugging Profile Manager   -     IYK2Z2T1

View Java Debugging Profile EJB      for $EXAMPLE

Bean (Enterprise Beans only)
==> example_bean_name
==>
==>
==>

Method (Enterprise Beans or Corba)
==> example_method_name
==>
==>
==>









PF1=Help 3=Exit 7=Back 12=Return
```

*Figure 11. The View Java debugging profile screen, showing example profile EJB*

The information displayed is described in "The Create Java debugging profile screen" on page 51.

**Related tasks**:

"Viewing a debugging profile with the 3270 interface" on page 44

**Related reference**:

"The List debugging profiles screen" on page 39

**Function keys for the View Java debugging profile screen:** The function keys for the "View Java debugging profile" screen are:

**PF1**
: Display the help screen

**PF3**
: End the debugging profile manager

**PF7**
Scroll forward
**PF8**
Scroll backward
**PF12**
Return to the "List debugging profiles" screen

## Setting the display device
### About this task

When you have created a debugging profile for a compiled language program, but before you can start debugging the application programs defined in the profile, you must specify the display device with which you will interact with the debugger. You can use:

a 3270 terminal
a debugging tool on a workstation.

The display device becomes associated with a debugging profile when you activate the profile, and remains associated with the profile until you make the profile inactive.

You can choose when you specify the display device:

- You can specify the display device before you make a profile active. The same display device will be associated with each profile that you subsequently make active.
- You can specify the display device when you make a profile active. The same display device will be associated with the profile that is made active. If you make more than one profile active at the same time (by selecting a number of profiles on the "List debugging profile" screen) the same display device will be associated with them all.

**Related reference**:

"The List debugging profiles screen" on page 39

"The Set LE debugging display device screen" on page 57

**Specifying the display device before you make a profile active:** Starting with the "List debugging profiles" screen, follow these steps:

1. Press PF9 to display the "Set LE debugging display device" screen.
2. Complete the details of the display device that you want to associate with the profile.
3. Type "No" in the "Display this panel on each LE profile activation" field.
4. Press ENTER. CICS saves the display device settings, and displays the "List debugging profiles" screen.

The "Set LE debugging display device" screen will not be displayed when you activate profiles; the settings you have supplied will be applied to all compiled language profiles when you activate them.

**Specifying the display device when you make a profile active:** Starting with the "List debugging profiles" screen, follow these steps:

1. Press PF9 to display the "Set LE debugging display device" screen.
2. Complete the details of the display device that you want to associate with the profile.
3. Type "Yes" (the default value) in the "Display this panel on each LE profile activation" field.

4. Press ENTER. CICS saves the display device settings, and displays the "List debugging profiles" screen.

CICS will display the "Set LE debugging display device" screen whenever you activate profiles for compiled language programs. The screen is initialized with the last settings you supplied. To change the settings:

1. Change any details of the display device that you want to associate with the set of profiles that is being activated.
2. Press ENTER. The display device settings are saved, and associated with the profile. CICS displays the "List debugging profiles" screen.

## The Set LE debugging display device screen

Use the "Set LE debugging display device" screen to specify the display device with which you will interact with the debugger.

```
CADP      -    CICS Application Debugging Profile Manager   -     IYK2Z2T1

Set LE Debugging Display Device

Debugging Display Device
Session Type             ==>  3270                          (3270,TCP)
3270 Display Terminal    ==>  TC15

TCP/IP Name Or Address
==>
==>
==>
==>
Port                     ==>  08000

Type of socket communication ==> Single                     (Single,Multiple)


Display this panel on LE profile activation ==> YES




Enter=Save and return PF1=Help 3=Exit 12=Cancel
```

*Figure 12. The CADP Set LE debugging display device screen*

**Related tasks**:

"Activating debugging profiles with the 3270 interface" on page 43

"Setting the display device" on page 56

**Related reference**:

"The List debugging profiles screen" on page 39

**Fields on the Set LE debugging display device screen:** The fields on the "Set LE debugging display device" screen are:

**Session type**
Specifies how you will interact with Debug Tool:

**3270**
You will interact with Debug Tool using a 3270 type terminal. Specify the terminal id in the display id field. This is the default value.

**TCP**
You will interact with Debug Tool using a debugging client on your

workstation. The client will communicate with Debug Tool using TCP/IP. Specify the port number at which the client listens for a connection in the port field.

**3270 display terminal**
When the session type is 3270, specify the terminal id of the terminal with which you will interact with Debug Tool. The default value is the id of the terminal at which you running CADP.

**Important:** The terminal specified here is not necessarily the terminal at which the transaction being debugged will run.

**TCP/IP name or address**
When the session type is TCP, specify the IP address or name of the host where the debugging client is running.

**Port**
When the session type is TCP, specify the port number at which the debugging client listens for a connection. Specify a value in the range 0 - 65535. The default is 8000.

**Type of socket communication**
When the session type is TCP, specifies whether the debugging client and debugging server will communicate using a single socket or more than one socket.

**Single**
Use a single socket for communication. This is the default value, and is the preferred value when you use a WebSphere Studio product as your debugging client.

**Multiple**
Use more than one socket for communication. You must specify this value when you use a VisualAge product as your debugging client.

**Display this panel on LE profile activation**
Specifies whether you want to display the "Set LE debugging display device" screen whenever you activate debugging profiles for compiled language programs:

**YES** Display the "Set LE debugging display device" screen whenever debugging profiles are activated. This is the default behavior.

**NO** Do not display the "Set LE debugging display device" screen whenever debugging profiles are activated. The display device that you specify will be associated with all the profiles that you activate.

**Function keys for the Set LE debugging display device screen:** The function keys for the "Set LE debugging display device" screen are:

**PF1**
Display the help screen

**PF3**
End the debugging profile manager

**PF12**
Return to the "List debugging profiles" screen

# Chapter 2. Starting and stopping CICS

This is information about how CICS starts and stops, the implications of different types of start and stop, and the actions you must take to handle them.

## What happens when you start CICS

When you start CICS, you start a process called **CICS system initialization**. This process must finish before you run any transactions.

CICS system initialization involves many activities, some of which are:

- Obtaining the required storage for CICS execution from the private area in the CICS address space, above and below the 16MB line.
- Setting up CICS system parameters for the run, as specified by the system initialization parameters.
- Loading and initializing the CICS domains according to the start option specified by the **START** system initialization parameter.
- Loading the CICS nucleus with the required CICS modules.
- Installing CICS resource definitions by:
  - Loading, from the CSD, the groups of resources specified by the **GRPLIST** system initialization parameter
  - Loading the control tables specified by system initialization parameters.
- Opening the data sets necessary for initialization, including any needed for backout if the previous run of your CICS region was not shut down normally.
- Opening BSAM sequential devices as required in the terminal control table (TCT).

If you are operating CICS with CICS recovery options, backout procedures can be used to restore recoverable resources to a logically consistent state. Backout occurs if you start CICS with START=AUTO and CICS detects that the previous shutdown was immediate or uncontrolled.

For background information about backout, and recovery and restart, see Troubleshooting for recovery processing.

### The types of CICS startup

CICS can start in any of the following ways.

*Table 2. CICS startup types and effect*

| Startup type | Effect |
|---|---|
| **Initial** | CICS starts with no reference to any system activity recorded in the CICS global catalog and system log from a previous run of CICS. For more information, see “CICS actions on an initial start” on page 60. |
| **Cold** | CICS starts with limited reference to any system activity recorded in the CICS global catalog and system log from a previous run of CICS. For more information, see “CICS actions on a cold start” on page 60. |

© Copyright IBM Corp. 1989, 2017

*Table 2. CICS startup types and effect (continued)*

| Startup type | Effect |
|---|---|
| **Warm** | CICS starts, after a normal shutdown, restoring CICS to the status it was in at the last normal CICS shutdown, except for some facilities that it initializes as for a cold start. CICS always restores the trace domain according to the system initialization parameters, and can restore other facilities depending on the COLD option of their associated system initialization parameters. For more information, see "CICS actions on a warm start" on page 61. |
| Emergency | CICS starts, after an abnormal shutdown, restoring recoverable resources to their committed states. For more information, see "CICS actions on an emergency restart" on page 64. |

When CICS is started, the type of startup (and therefore the actions it takes) depends primarily on the following:

- The value of the START system initialization parameter
- Two records in the CICS global catalog:
  - The recovery manager control record
  - The recovery manager autostart override record.

The values of other system initialization parameters also influence the actions taken on CICS startup. For information about the types of startup, the roles of the CICS catalogs, and the effect of the START system initialization parameter, see Controlling start and restart.

**Note:** You cannot explicitly request a warm or emergency restart. When selecting the type of start (using the START system initialization parameter), the choices are INITIAL, COLD, or AUTO. AUTO can result in a warm or an emergency restart; CICS itself determines which to use.

## CICS actions on an initial start

The CICS global catalog and system log are initialized, and all information in them is lost.

Resynchronization information for remote systems is not preserved, so damage might be done to distributed units of work. It should rarely be necessary to perform an initial start. Examples of times when an initial start is necessary are:

- When bringing up a new CICS system for the first time
- After a serious software failure, when the global catalog or system log has been corrupted.

## CICS actions on a cold start

In a cold start, initialization of CICS occurs with limited reference to any system activity recorded in the CICS catalogs. With the exception of resynchronization information for remote systems noted below, no system log or warm keypoint information is used from any previous run of CICS. Dump table entries from a previous run are also deleted in a cold start.

In a cold start:

- TERMINAL definitions are purged from the recovery file and from the catalog.
- Existing TYPETERM and MODEL definitions are purged from the catalog.
- PROGRAM definitions are purged from the recovery file and from the catalog.
- TRANSACTION and PROFILE definitions are purged from the global catalog.
- Transient data queue (TDQUEUE) definitions are purged from the catalog.
- File control records are purged from the catalog.
- Any program LIBRARY definitions that had been dynamically defined will be lost. Only the static DFHRPL concatenation will remain, together with any LIBRARY definitions in the grouplist specified at startup or installed via BAS at startup.
- Resource definition information is obtained as follows:
  - Tables specified by system initialization parameters, such as MCT=xx, are obtained from the program library.
  - Information in the groups in the list named by the GRPLIST system initialization parameter for *this* initialization is taken from the CICS system definition (CSD) file and merged with information from the program library.
  - Information in groups that have been defined or added to group lists is taken from the CSD.
- Resynchronization information relating to remote systems or to RMI-connected resource managers is preserved. The CICS system log is scanned during startup, and information regarding unit of work obligations to remote systems, or to non-CICS resource managers (such as DB2®) connected through the RMI, is preserved. (That is, any decisions about the outcome of local UOWs, needed to allow remote systems or RMI resource managers to resynchronize their resources, are preserved.)

  However, note that recovery information for remote systems connected by LU6.1 links, or for earlier releases of CICS systems connected by MRO is *not* preserved.
- The journal DFHLOG and DFHSHUNT entries in the catalog are used, and all other journals and journal models are purged.

## CICS actions on a warm start

A warm start restores certain elements of the CICS components that can be warm started to the status that was recorded in the warm keypoint of the previous normal shutdown.

A partial warm start is similar to a complete warm start, except that some selected CICS facilities are started cold, as specified in the system initialization parameters. Information is obtained for those facilities from the warm keypoint only if they are not specified for a cold start.

In a warm start:

- Resource definition information is obtained as follows:
  - Tables specified by system initialization parameters, such as **MCT**=xx, are obtained from the program library. Information contained in the warm keypoint of the previous run is used to update the information from the program library.
  - Information in the groups in the list named by the **GRPLIST** system initialization parameter for this initialization is ignored.
  - Information in the groups in the list named by the **GRPLIST** system initialization parameter for the previous initialization is obtained from the warm keypoint and the global catalog.

  - Information in groups that have been installed since the last cold start is obtained from the warm keypoint and the global catalog.
  - Information in groups that have been defined or added to group lists is taken from the CSD.
  - All dynamic LIBRARY resource definitions will be restored from the catalog, and the actual search order through the list of LIBRARY resources that was active at the time of the preceding shutdown will be preserved. If a library that is defined as critical is restored from the catalog and an error occurs in enabling the LIBRARY, such as one of the data sets in its concatenation no longer being available, a `Go` or `Cancel` message will be issued to allow the operator to choose whether to continue the CICS startup regardless, or to fail the startup. The message will be preceded by a set of messages providing information on any data sets which are not available. For non critical LIBRARY resources, this condition will not cause CICS startup to fail, but a warning message will be issued and the library will not be reinstalled.
  - Information about any autoinstalled terminal that has an automatic-initiate descriptor (AID) outstanding is retrieved from the global catalog.
- Selected fields from the CSA are restored from the warm keypoint, including:
  - Region exit time interval value
  - Runaway time interval value
  - Maximum number of tasks
  - High watermark number of the unit of recovery descriptor.
- The following pieces of information relating to logically recoverable, physically recoverable and non-recoverable intrapartition transient data queues are restored:
  - All data defining the queues. This information is restored from the global catalog, including trigger level information, ATI transaction IDs, ATI terminal IDs and so on.
  - All state-related data. This information is retrieved from the warm keypoint which was written to the log, including:
    - Record count
    - Read pointer value
    - Write pointer value
    - Information about whether or not a trigger transaction has been attached.

    All intrapartition transient data queues are installed as ENABLED. Trigger transactions are rescheduled if required.

    Extrapartition transient data queues are opened if OPEN=INITIAL is specified in the queue definition.
- The following FCT information is restored to what it was at the time of the warm shutdown, using information from the global catalog:
  - The ENABLED/DISABLED/UNENABLED status
  - The SERVREQ options (UPDATE, DELETE and so on)
  - Any alterations made to the DSNAME.
- Files defined as initially OPEN are opened irrespective of their other attributes. If the file state recovered during initialization is ENABLED or UNENABLED, the file becomes OPEN, ENABLED after the OPEN. If the file state recovered is DISABLED, the file becomes OPEN, DISABLED.
- Installed transaction and profile definitions are obtained from:
  - The groups specified in the **GRPLIST** system initialization parameter at the last cold start

  - The groups that have been installed since the last cold or emergency start.

  The following attributes of the installed transactions and profiles are restored from the warm keypoint:

  - ENABLED/DISABLED status
  - Transaction priority.
- Installed program and mapset definitions are obtained from these sources:
  - The groups specified in the **GRPLIST** system initialization parameters at the last cold start
  - The groups that have been installed **since** the last cold start or emergency restart
  - The changes (such as LPA-eligibility) made by CEMT or EXEC CICS SET PROGRAM commands in the last run.

  The ENABLED/DISABLED status of each installed program and mapset is restored from the warm keypoint. Directory information is obtained for each program and mapset during CICS initialization.
- The following TCT information is restored from the warm keypoint information:
  - Processing status (transaction, transceive, input, or receive)
  - Service status (INSERVICE or OUTSERVICE)
  - Extended attributes supported (color, programmed symbols, and so on)
  - Partition support
  - Magnetic-stripe-reader support
  - Outboard formatting support
  - Coded graphic character set identifiers
  - APL/TEXT keyboard.

  If any outstanding work was scheduled for an autoinstalled terminal at the last warm shutdown, the terminal entry is recovered. (Terminal entries for autoinstalled terminals with no work outstanding are deleted at shutdown.)
- The following auxiliary temporary storage information is restored from the warm keypoint:
  - All data in the auxiliary temporary storage queues
  - The temporary storage use map.
- Interval control elements (ICEs) for outstanding **START TRANSID** commands are restored from the warm keypoint.
- The BMS logical messages that were created by the functions listed below but have not yet been viewed by the terminal operator are restored:
  - Message switching transaction (CMSG).
  - ROUTE command.
  - **SEND MAP ACCUM** and **SEND TEXT ACCUM** commands, except for those messages terminated by **SEND PAGE** without specifying RELEASE or RETAIN. In those cases, the message might already have been viewed by the operator, but can be viewed again following the warm start.
- All unit of recovery descriptors (APPC log name, APPC resynchronization, and external resource manager) are restored from the warm keypoint, together with any associated deferred work elements (DWEs).
- The STORECLOCK value is restored from the warm keypoint.
- The intervals at which statistics were collected and status and the logical end-of-day time are restored from the global catalog.

- The monitoring status, class status and monitoring control table suffix are restored from the global catalog.
- Transaction and system dump table options are held in the global catalog and reapplied at a warm start.
- Journals and journal models are restored from the catalog.
- The version of Java in use for the CICS region (supported by the IBM 64-bit SDK for z/OS, Java Technology Edition) is restored from the global catalog.

## CICS actions on an emergency restart

A CICS system that operates on resources, such as files, that have been defined by the installation to be *recoverable*, records changes to those resources in the CICS system log.

If the CICS system fails, the system log at the time of failure should typically contain records of changes made by tasks that have not completed ('in-flight' tasks) and by others that have completed.

Following an abnormal termination, Recovery Manager collects all of the log records pertaining to in-flight tasks. It acquires locks on any records that they updated and restores the tasks as shunted UOWs, to be backed out after initialization is complete.

### CICS-z/OS Communications Server actions after an emergency restart

When LU-LU sessions are re-established after an emergency restart (and subsequent processing), CICS participates in a resynchronization protocol with logical units to discover if any messages, in either direction, were lost when CICS was terminated.

The logical units for which resynchronization is required will have been marked in the TCTTEs. Resynchronization is *not* attempted in the following cases:

- If the terminal was acquired by a master terminal operation specifying COLDACQ.
- If the terminal was acquired with the EXEC CICS SET TERMINAL ACQSTATUS(COLDACQ) command.
- If the session is a pipeline session.
- If the TCTTE is marked to perform a cold start of the session by the TCT assembly process. This is done for terminals such as 3270 terminals that do not support the set and test sequence number (STSN) command.

**Note:** If the previous session abended, the use of COLDACQ overrides CICS integrity control. This could lead to data integrity problems. Also, you should check the CSMT log for an activity keypoint after the restart of a session following a CICS failure. If there is no activity keypoint, you should issue COLDACQ again after the next emergency restart.

For each logical unit that *does* require resynchronization, CICS issues an STSN command that notifies the logical unit of the sequence numbers known to CICS—that is, those numbers that backout processing placed in the TCTTE. The logical unit can compare these sequence numbers with those that it has logged for itself, and can thus determine if any messages were lost.

- If an *input* message was lost, the logical unit should retransmit it to CICS.

- If an *output* message was lost, CICS retransmits the message from the resend slot and, in so doing, deletes the resend slot.

The message remains in the resend slot if CICS does *not* retransmit it. This occurs if the resynchronization process shows that the output message was not lost, or if the logical unit does not support the STSN command; the 3270 is in this category.

## CICS startup and the z/OS Communications Server session

In an SNA network, the session between CICS and the z/OS Communications Server for SNA is started automatically if the z/OS Communications Server is started before CICS.

If the z/OS Communications Server is not active when you start CICS, you receive the following messages:

```
F vtamname,USERVAR,ID=generic-applid,VALUE=specific-applid
+DFHSI1589D 'applid' VTAM is not currently active.
+DFHSI1572 'applid'  Unable to OPEN VTAM ACB - RC=xxxxxxxx, ACB CODE=yy.
```

If you receive messages DFHSI1589D and DFHSI1572, you can start the CICS-z/OS Communications Server session manually when the Communications Server is eventually started, by using the **CEMT SET VTAM OPEN** command from a supported MVS console or a non-SNA LU.

If the z/OS Communications Server is active, but CICS still cannot open the SNA ACB because the Communications Server does not recognize the CICS APPLID, you receive the following messages:

```
F vtamname,USERVAR,ID=generic-applid,VALUE=specific-applid
+DFHSI1592I 'applid' CICS applid not (yet) active to VTAM.
+DFHSI1572  'applid' Unable to OPEN VTAM ACB - RC=00000008, ACB CODE=5A.
```

This could be caused by an error in the value of APPLID operand, in which case you must correct the error and restart CICS. For information about other causes and actions, see CICS messages.

## End of CICS startup

Whichever type of startup is performed, when the DFHSI1517 is displayed on the operating system console, CICS is ready to process terminal requests.

```
DFHSI1517 - 'applid':  Control is being given to CICS.
```

where *applid* is the value of the specific APPLID system initialization parameter.

When the startup process is completed, users are able to enter transactions from any terminals that are connected to CICS.

Even though DFHSI1517 signals the end of system initialization, the region might not be fully ready to receive work. This is because in some cases initialization continues after this message. For example, some bundle-defined resources are being installed asynchronously; so resources such as JVMSERVERs required for some applications have not completed initialization, even though the TCP/IP listener is open and accepting work. This could cause transactions to abend. For web services, a pipeline scan might not have completed, so use of the web service before the scan completes will fail.

You can allow CICS to have a warm-up process by setting the **WLMHEALTH** system initialization parameter. If the z/OS Workload Manager health service is active in the region, when the message DFHSI1517 is returned, the warm-up process is

started. Then CICS adjusts the region's z/OS WLM health indicator to control flow of work into the region until the region is fully capable to process work. For more information about the CICS warm-up process, see CICS warm-up and cool-down by use of z/OS Workload Manager health service.

# Managing CICS shutdown

This section describes how to shut down CICS and processing of system shutdown.

## Before you begin

Before CICS shutdown, you can initiate a system cool-down to gradually stop flow of work into the region. To do so, you must set the WLMHEALTH system initialization parameter to activate the z/OS Workload Manager Health service in the CICS region. To initiate a system cool-down, you can issue a **SET WLMHEALTH OPENSTATUS(CLOSE)** command or use the CICS Explorer. For more information, see "CICS warm-up and cool-down by use of z/OS Workload Manager health service" on page 71.

# The types of CICS shutdown

There are three types of CICS system shutdown - normal, immediate, and uncontrolled.

## Normal shutdown

In a normal shutdown, CICS performs a controlled sequence of operations that leave the system in a well-defined state. Existing tasks are allowed to finish. The following events can cause a normal shutdown of CICS:

- Using the **CEMT PERFORM SHUTDOWN** transaction
- Using the **EXEC CICS PERFORM SHUTDOWN** command

**Note:** During shutdown the trace subtask TCB will be detached. This will cause ABEND13E. This is normal processing, and is to be expected.
After a normal shutdown, it is possible to warm start CICS.

## Immediate shutdown

In an immediate shutdown, CICS remains in overall control, but it does a minimum amount of processing so the system can terminate rapidly. Existing tasks are not allowed to finish, and could abend. If the CESD default shutdown transaction is enabled, existing tasks are given a short time to finish before they are purged.

The following events can cause an immediate shutdown of CICS:

- Using the **CEMT PERFORM SHUTDOWN IMMEDIATE** transaction
- Using the **EXEC CICS PERFORM SHUTDOWN IMMEDIATE** command
- A CICS system abend
- A program check

**Note:** During shutdown the trace subtask TCB will be detached. This will cause ABEND13E. This is normal processing, and is to be expected.
After an immediate shutdown, you must perform an emergency restart or a cold start.

### Uncontrolled shutdown

In an uncontrolled shutdown, CICS is not given the chance to do any processing after the event causing it to terminate has occurred.

The following events can cause an uncontrolled shutdown of CICS:

- Power® failure
- Machine check
- Operating system failure

In each case, CICS cannot perform any shutdown processing. In particular, CICS does *not* write a warm keypoint or a warm-start-possible indicator to the global catalog.

To preserve data integrity, the next initialization of CICS **must** be an emergency restart. If you specify START=AUTO on the next initialization of CICS, there will be an emergency restart.

## What happens during CICS shutdown

Normal shutdown involves quiesce processing. In comparison, immediate shutdown is accomplished by termination processing.

### CICS actions on a normal shutdown (PERFORM SHUTDOWN)

Normal shutdown is initiated by the master terminal operator or by an application program. There are two stages in a normal shutdown.

#### First stage of normal shutdown

During the first stage of CICS normal shutdown, all terminals are active and all CICS facilities are available.

The following actions take place concurrently:

- Message DFHTM1715 is issued to the console and the master terminal user to inform the operator that CICS is terminating.
- Tasks that already exist will complete. Long running tasks, such as conversational tasks, must end before this stage of shutdown can complete.
- Tasks to be automatically initiated will run, if they can start before the second stage.
- Any user-written programs listed in the first part of the shutdown program list table (PLT) are run sequentially.
- The Front End Programming Interface (FEPI) is requested to shut down.
- The terminal that initiated the shutdown, if any, is detached. This allows the operator to start any further tasks that might be required, or to purge any tasks.

  A new task is allowed to start only if it has been defined as SHUTDOWN(ENABLED) in its TRANSACTION resource definition, or, for a transaction started as a result of terminal input, if the transaction identifier is listed in the current transaction list table (XLT). The XLT list of transactions restricts the tasks that can be started by terminals and allows the system to shut down in a controlled manner. The current XLT is the one specified by the **XLT** system initialization parameter, which can be overridden by the XLT option of the CEMT or EXEC CICS PERFORM SHUTDOWN command.

Certain CICS-supplied transactions are, however, allowed to start whether or not their code is listed in the XLT. These transactions are CEMT, CESF, CLS1, CLS2, CSAC, CSTE, and CSNE.

**Note:** You should not change the SHUTDOWN(ENABLED) attribute of the resource definitions for these transactions, otherwise CICS might not shut down successfully.

- A request is issued to all interregion communication (IRC) activity.
- Terminal control is requested to ignore all further input.
- Unless SDTRAN=NO or NOSDTRAN was specified, the shutdown task starts the specified shutdown transaction (default is CESD). CESD manages the purging of long-running user tasks.
- CLSDST requests are issued for all z/OS Communications Server terminals.
- The termination task waits for all terminal activity to cease, before entering the second stage of shutdown.

The first shutdown stage is complete when the last of the programs specified in the first part of the shutdown PLT has run and all user tasks are complete.

### Second stage of normal shutdown

During the second stage of shutdown, terminals are not active, and no new tasks are allowed to start.

The following processing takes place:

1. User-written programs listed in the second part of the shutdown PLT (if any) are executed sequentially. These programs cannot communicate with terminals, or make any request that would cause a new task to start.
2. All currently open CICS files are now closed.
3. The transient data CI buffer and the temporary storage buffers are flushed.
4. CICS writes the following information to the global catalog:
   - A warm keypoint. This contains information that is used to restore the operating environment during a subsequent warm start.
   - A warm-start-possible indicator. This status applies on the next initialization of CICS if START=AUTO is specified.
5. Transient data is terminated.
6. A dump is taken, if one is required.
7. The local and global catalogs are closed.
8. The following message is issued:

   ```
   DFHKE1799 applid TERMINATION OF CICS IS COMPLETE
   ```

9. CICS completes some internal processing, then returns control to MVS.

## CICS actions on an immediate shutdown (PERFORM SHUTDOWN IMMEDIATE)

During immediate shutdown of CICS, possibly requested by the master terminal operator or an application program, processing is different from a normal shutdown in the following important ways.

- User tasks are not guaranteed to complete for any kind of shutdown. They are just given less time for immediate shutdown before being purged.
- None of the programs listed in the shutdown PLT are run.

- CICS does *not* write a warm keypoint or a warm-start-possible indicator to the global catalog.
- CICS does not close files defined to CICS file control.

To preserve data integrity, the next initialization of CICS **must** be an emergency restart. If the next initialization of CICS specifies START=AUTO, there will be an emergency restart.

The processing involved in immediate shutdown is described as CICS system termination processing. (In comparison, normal shutdown involves quiesce processing.)

Unlike processing, controls are not exercised to ensure that resources and services remain available as long as they are needed. One consequence of this is that transaction and CICS system abends can occur during immediate shutdown. Thus, if a task tries to use a resource that has already been terminated, the task abends. Then dynamic transaction backout is invoked, and that might also fail because it could also try to use a resource that has been terminated.

In addition, if CICS system termination processing is delayed significantly, tasks in the system waiting for input from terminals that are no longer available are likely to extend beyond the period for deadlock timeout specified in the DTIMOUT option of the TRANSACTION definition.

### First stage of immediate shutdown

During the first stage of an immediate shutdown, the following processes take place:

1. The system termination task drives the collection of termination statistics.
2. If there is a terminal associated with the event that caused the immediate shutdown, a message is sent to inform the operator that CICS is terminating.
3. If the shutdown request has arrived by transaction routing, the associated terminal is freed.
4. Terminal input is no longer accepted.
5. The Front End Programming Interface (FEPI) is requested to shut down immediately. Unless SDTRAN=NO or NOSDTRAN was specified, the shutdown task starts the specified shutdown transaction (the default is CESD). CESD manages the purging of long-running user tasks.

### Second stage of immediate shutdown

During the second stage of an immediate shutdown, the following processing takes place:

1. Transient data is terminated.
2. A dump is taken, if requested.
3. Interregion sessions are terminated.
4. If CICS is signed on to the CICS availability manager (CAVM), a “signoff abnormal” request is made from CAVM.
5. The local catalog and global catalog are left to be closed by the operating system.
6. The following message is issued:

   `DFHKE1799 applid TERMINATION OF CICS IS COMPLETE`
7. CICS completes some internal processing, then returns control to MVS.

# Shutting down CICS normally

You can use **CEMT PERFORM SHUTDOWN** to initiate a normal shutdown.

## Before you begin

- Make sure that all pipes (sessions) that are in use for the external call interface have been closed; otherwise, CICS cannot complete a normal shutdown.
- Before CICS shutdown, you can initiate a system cool-down to gradually stop flow of work into the region. To do so, you must set the WLMHEALTH system initialization parameter to activate the z/OS Workload Manager Health service in the CICS region. To initiate a system cool-down, you can issue a **SET WLMHEALTH OPENSTATUS(CLOSE)** command or use the CICS Explorer. For more information, see "CICS warm-up and cool-down by use of z/OS Workload Manager health service" on page 71.

## Procedure

Use the command **CEMT PERFORM SHUTDOWN** to shut down CICS normally. The command can be issued at the system console or the master terminal. When you use this command, CICS responds directly by issuing the following messages at the console:

```
DFHTM1715 CICSITH1 CICS is being quiesced by userid IVPUSER
                   in transaction CEMT at netname IG2S2CA8.
DFHDM0102I applid CICS is quiescing.
```

Message DFHTM1715 is also issued to the master terminal, to inform the operator that CICS is terminating.

## Results

If the normal shutdown is successful, CICS issues the following message at the console:

## Example

For example, the following sequence of messages is issued on a normal shutdown of the CICS TOR, CICSHTH1:

```
13.04.37 JOB08579  +DFHTM1715 IYK4ZEE1 CICS is being quiesced by userid CICSUSER
                              in transaction CEMT at netname IG2S66B9.
13.04.37 JOB08579  +DFHDM0102I IYK4ZEE1 CICS is quiescing.
13.04.37 JOB08579  +DFHMN0115I IYK4ZEE1 CICS Server z/OS WLM Health percentage is now 0.
13.04.37 JOB08579  +DFHCESD  IYK4ZEE1 SHUTDOWN ASSIST TRANSACTION CESD STARTING.
                              SHUTDOWN IS NORMAL.
13.04.37 JOB08579  +DFHTM1781 IYK4ZEE1 CICS shutdown cannot complete because some
                              non-system user tasks have not terminated.
13.06.37 JOB08579  +DFHCESD  IYK4ZEE1 THERE ARE NOW 0002 TASKS STILL IN THE SYSTEM.
13.06.54 JOB08579  +DFHDU0303I IYK4ZEE1 Transaction Dump Data set DFHDMPA closed.
13.06.54 JOB08579  +DFHCESD  IYK4ZEE1 PURGING TRANID CECI, TERMID 66B8, USERID CICSUSER,
                              TASKNO 000026
13.06.56 JOB08579  +DFHCESD  IYK4ZEE1 THERE ARE NOW 0001 TASKS STILL IN THE SYSTEM.
13.06.59 JOB08579  +DFHTM1782I IYK4ZEE1 All non-system tasks have been successfully
                              terminated.
13.06.59 JOB08579  +DFHZC2305I IYK4ZEE1 Termination of VTAM sessions beginning
13.07.01 JOB08579  +DFHZC2316 IYK4ZEE1 VTAM ACB is closed
13.07.03 JOB08579  +DFHRM0204 IYK4ZEE1 There are no indoubt, commit-failed or
                              backout-failed UOWs.
13.07.04 JOB08579  +DFHRM0130 IYK4ZEE1 Recovery manager has successfully quiesced.
13.07.07 JOB08579  +DFHKE1799 IYK4ZEE1 TERMINATION OF CICS IS COMPLETE.
```

**Note:** VTAM® is now the z/OS Communications Server (for SNA or IP).

## Shutting down CICS immediately

You can use **CEMT PERFORM SHUTDOWN IMMEDIATE** to initiate an immediate shutdown.

### Procedure

To shut down CICS immediately, use the command **CEMT PERFORM SHUTDOWN IMMEDIATE**. You can use this command at the system console or the master terminal. When you use this command, CICS responds directly by issuing the DFHTM1703 message at the console. Message DFHTM1703 is also issued to the master terminal, to inform the operator that CICS is terminating.

### Results

If the CICS shutdown is successful, CICS issues the following message at the console:

```
DFHKE1799 applid TERMINATION OF CICS IS COMPLETE
```

### Example

For example, the following sequence of messages was issued on an immediate shutdown of the CICS TOR, CICSHTH1:

```
16:15:59 . F CICSHTH1,CEMT PERF SHUT IMMED
15.05.55 . +DFHTM1703 CICSITH1 CICS is being quiesced by userid
                      IVPUSER in transaction CEMT at terminal SAMA
16.15.59 . +DFHTM1701 CICSHTH1 CICS is being terminated by operator
                      at terminal CON1
16.16.01 . +DFHDU0303I CICSHTH1 Transaction Dump Data set DFHDMPA closed.
16.16.01 . +DFHKE1799 CICSHTH1  TERMINATION OF CICS IS COMPLETE.
```

## Shutdown command options

You can specify any of the following shutdown options on the command, without affecting the type of shutdown performed.

**Option Effect**

**DUMP**
CICS produces a dynamic storage dump after shutdown has completed.

**PLT(xx)**
CICS runs programs in the PLT, DFHPLTxx, during shutdown.

**XLT(xx)**
Only those transactions listed in the XLT, DFHXLTxx, can be started after the SHUTDOWN command, and before shutdown has completed.

# CICS warm-up and cool-down by use of z/OS Workload Manager health service

By use of the z/OS Workload Manager (z/OS WLM) health service, you can allow CICS to have a warm-up process after the end of system initialization to control flow of work into the region until the region is fully ready to receive work. You can also allow CICS to have a cool-down process before system shut-down to limit flow of work into the region.

## Benefits

Using the z/OS WLM health service to control flow of work into the CICS region after the completion of system initialization and before CICS shutdown has the following benefits:

**A system warm-up process can mitigate problems that can be caused by work flowing into a CICS region just after the completion of system initialization.**
Even though DFHSI1517 signals the end of system initialization, CICS might still not be fully ready to receive and process work. This is because in some cases initialization continues after this message. For example, some bundle-defined resources are being installed asynchronously; so resources such as JVMSERVERs required for some applications have not completed initialization, even though the TCP/IP listener is open and accepting work. This could cause transactions to abend. For web services, a pipeline scan might not have completed, so use of the web service before the scan completes will fail.

A system warm-up process can be used to control work flowing into CICS when the region is not yet running at optimum performance.

**You have a lot of flexibility over control of the warm-up or cool-down process.**
How long a CICS region needs to warm up before it is fully healthy depends on the types of applications being processed and the set of resources they employ. You can control the warm-up or cool-down process based on the need of each CICS region.

## How to activate and set up the z/OS WLM health service

The z/OS WLM health service is enabled by default. If the service is disabled, to activate this service for a CICS region, you must set the **WLMHEALTH** system initialization parameter. In **WLMHEALTH**, you must specify an interval value and a health adjustment value.

When the region is operating, you can change the interval value and health adjustment value by using the **SET WLMHEALTH** SPI command, **CEMT SET WLMHEALTH**, the CICSPlex SM web user interface, or the CICS Explorer.

## What happens during CICS warm-up or cool-down

When the z/OS WLM health service is active in a CICS region, CICS informs the z/OS Workload Manager of the health state of the region. The region's z/OS WLM *health indicator* is an integer in the range 1 - 100. A value of 100 means that the region is fully capable to do work without any health problems whereas 0 means that it is not able to do any work.

**CICS warm-up process**
During early CICS initialization, the region's health indicator is set to zero. When the DFHSI1517 `Control is being given to CICS` message is returned, the CICS warm-up process starts. At each interval, CICS calls the z/OS Workload Manager Health (IWM4HLTH) API to increment the region's health indicator from zero, by a specified adjustment value, until the health indicator reaches 100.

If the interval value on the **WLMHEALTH** parameter is set to zero, the health indicator of the region remains at zero. To initiate a warm-up process for

this region, you must change the interval to a non zero value and set WLMHEALTH open by, for example, issuing a **SET WLMHEALTH** SPI command.

**CICS cool-down process**
A cool-down process can be initiated by issuing a **SET WLMHEALTH CLOSE** SPI command. At each interval, CICS calls the z/OS Workload Manager Health (IWM4HLTH) API to decrement the region's health indicator from 100, by the adjustment value, until the health indicator is zero.

### How use of z/OS WLM health service affects CICS

The health state of a CICS region affects how favorable the region is a target for connections. CICS and CICSPlex SM components such as the CICS-MQ adapter react to the region's health state.

**TCP/IP**
When TCP/IP is configured, the health indicator will feed into the WLM server recommendations to sysplex distributor, affecting when connections are established with a CICS region.

**MQMONITOR**
Changes in the region's health state can have an effect on when MQMONITORs with attribute AUTOSTART(YES) are started or stopped. When the health indicator is less than 100, all started MQMONITORs are subject to a throttle limiting the messages that a single MQMONITOR can process. For details, see Effect of z/OS Workload Manager Health service on MQMONITORs.

**CICSPlex SM**
CICSPlex SM workload distribution takes account of the region's z/OS WLM health state in the routing algorithm. When CICSPlex SM Workload Manager is deciding where to route work, a region with a health indicator of zero is less favorable than a region with a non-zero health indicator. For details, see Effect of the z/OS WLM health service on CPSM workload routing.

### How to initiate CICS warm-up or cool-down

If set up correctly, a CICS warm-up process starts automatically at the end of system initialization.

You can also initiate a CICS warm-up or cool-down process using the **SET WLMHEALTH** SPI command, **CEMT SET WLMHEALTH**, the CICSPlex SM web user interface, or the CICS Explorer.

## Initiating a CICS system warm-up

If configured correctly, the CICS warm-up process starts when the DFHSI1517 is returned at the end of system initialization. You can also initiate a CICS warm-up process by issuing an **EXEC CICS SET WLMHEALTH** command in a CICS application, using the CEMT transaction, using the CICSPlex SM web user interface, or using the CICS Explorer.

### Before you begin

The z/OS WLM health service must be active for the CICS region. This service is enabled by default. If it is disabled, you can activate the service by setting the

**WLMHEALTH** system initialization parameter to a value other than OFF. In **WLMHEALTH**, you must specify an interval value and a health adjustment value.

### About this task

During early CICS initialization, the region's health indicator is set to zero. If the interval value on the **WLMHEALTH** parameter is set to a non zero value, when the DFHSI1517 `Control is being given to CICS` message is returned, the CICS warm-up process starts. However, if the interval value is zero, the health indicator of the region remains at zero. To initiate a warm-up process for this region, you must change the interval to a non zero value and then initiate the warm-up process.

### Procedure

- To initiate a CICS warm-up process in a CICS application using the SET WLMHEALTH command:
  - If you want to change the interval or adjustment value, issue the following command. If the interval value on the **WLMHEALTH** parameter is set to zero, you must set the interval to a non zero value.

    ```
    EXEC CICS SET WLMHEALTH INTERVAL(value) ADJUSTMENT(value)
    ```
  - To initiate the warm-up process, issue this command:

    ```
    EXEC CICS SET WLMHEALTH OPENSTATUS(OPEN)
    ```
- To initiate a CICS warm-up process using the CEMT transaction:

  **Note:** For details of how to start and use the CEMT transaction, see CEMT - master terminal.

  1. On the CICS command line, enter the command `CEMT SET WLMHEALTH`.
  2. Optional: To change the interval value, overtype the value in the INTERVAL field.
  3. Optional: To change the adjustment value, overtype the value in the ADJUSTMENT field.
  4. To initiate a CICS warm-up process, overtype the value CLOSE with `OPEN`.
  5. Press Enter.
- To initiate a warm-up process from the CICSPlex SM web user interface, use the MVS workload management (MVSWLM) view. To navigate to this view from the main menu, click **CICS operations views** > **CICS region operations views** > **MVS workload management**.

  You can also change the interval value and adjustment value in the MVS workload management (MVSWLM) view.
- To initiate a warm-up process from the CICS Explorer, follow the instruction available with the CICS Explorer documentation.

### Results

At each interval, CICS calls the z/OS Workload Manager Health (IWM4HLTH) API to increment the region's health indicator from zero, by the specified adjustment value, until the health indicator reaches 100. Then, the warm-up process is complete.

## Initiating a CICS system cool-down

Before CICS shut-down, you can initiate a CICS cool-down process by issuing an **EXEC CICS SET WLMHEALTH** command in a CICS application, using the CEMT

transaction, using the CICSPlex SM web user interface, or using the CICS Explorer. With the cool-down process, you can limit flow of work into a CICS region before system shut-down.

## Before you begin

The z/OS WLM health service must be active for the CICS region. This service is enabled by default. If it is disabled, you can activate the service by setting the **WLMHEALTH** system initialization parameter to a value other than OFF. In **WLMHEALTH**, you must specify an interval value and a health adjustment value.

## About this task

If you issue **SET WLMHEALTH IMMCLOSE**, the region's health indicator is set to zero immediately. There is no cool-down process with the **IMMCLOSE** option.

## Procedure

- To initiate a CICS cool-down process in a CICS application using the SET WLMHEALTH command:
  - If you want to change the interval or adjustment value, issue the following command:

    ```
    EXEC CICS SET WLMHEALTH INTERVAL(value) ADJUSTMENT(value)
    ```
  - To initiate the cool-down process, issue this command:

    ```
    EXEC CICS SET WLMHEALTH OPENSTATUS(CLOSE)
    ```
- To initiate a CICS cool-down process using the CEMT transaction:

  **Note:** For details of how to start and use the CEMT transaction, see CEMT - master terminal.

  1. On the CICS command line, enter the command `CEMT SET WLMHEALTH`.
  2. Optional: To change the interval value, overtype the value in the INTERVAL field.
  3. Optional: To change the adjustment value, overtype the value in the ADJUSTMENT field.
  4. To initiate a CICS cool-down process, overtype the value OPEN with `CLOSE`.
  5. Press Enter.
- To initiate a cool-down process from the CICSPlex SM web user interface, use the MVS workload management (MVSWLM) view. To navigate to this view from the main menu, click **CICS operations views** > **CICS region operations views** > **MVS workload management**.

  You can also change the interval value and adjustment value in the MVS workload management (MVSWLM) view.
- To initiate a cool-down process from the CICS Explorer, follow the instruction available with the CICS Explorer documentation.

## Results

At each interval, CICS calls the z/OS Workload Manager Health (IWM4HLTH) API to decrement the region's health indicator from 100, by the specified adjustment value, until the health indicator is zero. Then, the cool-down process is complete.

# Chapter 3. Administering restart and recovery

Before you begin to plan and implement resource recovery in CICS, you should understand the concepts involved, including units of work, logging and journaling.

## Units of work

When resources are being changed, there comes a point when the changes are complete and do not need backout if a failure occurs later. The period between the start of a particular set of changes and the point at which they are complete is called a *unit of work* (UOW). The unit of work is a fundamental concept of all CICS backout mechanisms.

From the application designer's point of view, a UOW is a sequence of actions that needs to be complete before any of the individual actions can be regarded as complete. To ensure data integrity, a unit of work must be atomic, consistent, isolated, and durable.

The CICS recovery manager operates with units of work. If a transaction that consists of multiple UOWs fails, or the CICS region fails, committed UOWs are not backed out.

A unit of work can be in one of the following states:

- Active (in-flight)
- Shunted following a failure of some kind
- Indoubt pending the decision of the unit of work coordinator.
- Completed and no longer of interest to the recovery manager

### Shunted units of work

A shunted unit of work is one awaiting resolution of an indoubt failure, a commit failure, or a backout failure. The CICS recovery manager attempts to complete a shunted unit of work when the failure that caused it to be shunted has been resolved.

A unit of work can be unshunted and then shunted again (in theory, any number of times). For example, a unit of work could go through the following stages:

1. A unit of work fails indoubt and is shunted.
2. After resynchronization, CICS finds that the decision is to back out the indoubt unit of work.
3. Recovery manager unshunts the unit of work to perform backout.
4. If backout fails, it is shunted again.
5. Recovery manager unshunts the unit of work to retry the backout.
6. Steps 4 and 5 can occur several times until the backout succeeds.

These situations can persist for some time, depending on how long it takes to resolve the cause of the failure. Because it is undesirable for transaction resources to be held up for too long, CICS attempts to release as many resources as possible while a unit of work is shunted. This is generally achieved by abending the user task to which the unit of work belongs, resulting in the release of the following:

- Terminals
- User programs

© Copyright IBM Corp. 1989, 2017

- Working storage
- Any LU6.2 sessions
- Any LU6.1 links
- Any MRO links

The resources CICS retains include:

- Locks on recoverable data. If the unit of work is shunted indoubt, all locks are retained. If it is shunted because of a commit- or backout-failure, only the locks on the failed resources are retained.
- System log records, which include:
  - Records written by the resource managers, which they need to perform recovery in the event of transaction or CICS failures. Generally, these records are used to support transaction backout, but the RDO resource manager also writes records for rebuilding the CICS state in the event of a CICS failure.
  - CICS recovery manager records, which include identifiers relating to the original transaction such as:
    - The transaction ID
    - The task ID
    - The CICS terminal ID
    - The z/OS Communications Server SNA LUNAME
    - The user ID
    - The operator ID.

## Locks

For files opened in RLS mode, VSAM maintains a single central lock structure using the lock-assist mechanism of the MVS coupling facility. This central lock structure provides sysplex-wide locking at a record level. Control interval (CI) locking is not used.

The locks for files accessed in non-RLS mode, the scope of which is limited to a single CICS region, are file-control managed locks. Initially, when CICS processes a read-for-update request, CICS obtains a CI lock. File control then issues an ENQ request to the enqueue domain to acquire a CICS lock on the specific record. This enables file control to notify VSAM to release the CI lock before returning control to the application program. Releasing the CI lock minimizes the potential for deadlocks to occur.

For coupling facility data tables updated under the locking model, the coupling facility data table server stores the lock with its record in the CFDT. As in the case of RLS locks, storing the lock with its record in the coupling facility list structure that holds the coupling facility data table ensures sysplex-wide locking at record level.

For both RLS and non-RLS recoverable files, CICS releases all locks on completion of a unit of work. For recoverable coupling facility data tables, the locks are released on completion of a unit of work by the CFDT server.

### Active and retained states for locks

CICS supports active and retained states for locks.

When a lock is first acquired, it is an active lock. It remains an active lock until successful completion of the unit of work, when it is released, or is converted into a retained lock if the unit of work fails, or for a CICS or SMSVSAM failure:

- If a unit of work fails, RLS VSAM or the CICS enqueue domain continues to hold the record locks that were owned by the failed unit of work for recoverable data sets, but converted into retained locks. Retaining locks ensures that data integrity for those records is maintained until the unit of work is completed.
- If a CICS region fails, locks are converted into retained locks to ensure that data integrity is maintained while CICS is being restarted.
- If an SMSVSAM server fails, locks are converted into retained locks (with the conversion being carried out by the other servers in the sysplex, or by the first server to restart if all servers have failed). This means that a UOW that held active RLS locks will hold retained RLS locks following the failure of an SMSVSAM server.

Converting active locks into retained locks not only protects data integrity. It also ensures that new requests for locks owned by the failed unit of work do not wait, but instead are rejected with the LOCKED response.

## Synchronization points

The end of a UOW is indicated to CICS by a synchronization point, usually abbreviated to syncpoint.

A syncpoint arises in the following ways:

- Implicitly at the end of a transaction as a result of an **EXEC CICS RETURN** command at the highest logical level. This means that a UOW cannot span tasks.
- Explicitly by **EXEC CICS SYNCPOINT** commands issued by an application program at appropriate points in the transaction.
- Implicitly through a DL/I program specification block (PSB) termination (TERM) call or command. This means that only one DL/I PSB can be scheduled within a UOW.

  Note that an explicit **EXEC CICS SYNCPOINT** command, or an implicit syncpoint at the end of a task, implies a DL/I PSB termination call.
- Implicitly through one of the following CICS commands:
  - **EXEC CICS CREATE TERMINAL**
  - **EXEC CICS CREATE CONNECTION COMPLETE**
  - **EXEC CICS DISCARD CONNECTION**
  - **EXEC CICS DISCARD TERMINAL**
- Implicitly by a program called by a distributed program link (DPL) command if the SYNCONRETURN option is specified. When the DPL program terminates with an **EXEC CICS RETURN** command, the CICS mirror transaction takes a syncpoint.

It follows from this that a unit of work starts:

- At the beginning of a transaction
- Whenever an explicit syncpoint is issued and the transaction does not end
- Whenever a DL/I PSB termination call causes an implicit syncpoint and the transaction does not end
- Whenever one of the following CICS commands causes an implicit syncpoint and the transaction does not end:
  - **EXEC CICS CREATE TERMINAL**

- **EXEC CICS CREATE CONNECTION COMPLETE**
- **EXEC CICS DISCARD CONNECTION**
- **EXEC CICS DISCARD TERMINAL**

A UOW that does not change a recoverable resource has no meaningful effect for the CICS recovery mechanisms. Nonrecoverable resources are never backed out.

A unit of work can also be ended by backout, which causes a syncpoint in one of the following ways:

- Implicitly when a transaction terminates abnormally, and CICS performs dynamic transaction backout
- Explicitly by **EXEC CICS SYNCPOINT ROLLBACK** commands issued by an application program to backout changes made by the UOW.

## Examples of synchronization points

In Figure 13, task A is a nonconversational (or pseudoconversational) task with one UOW, and task B is a multiple UOW task (typically a conversational task in which each UOW accepts new data from the user). The figure shows how UOWs end at syncpoints. During the task, the application program can issue syncpoints explicitly, and, at the end, CICS issues a syncpoint.

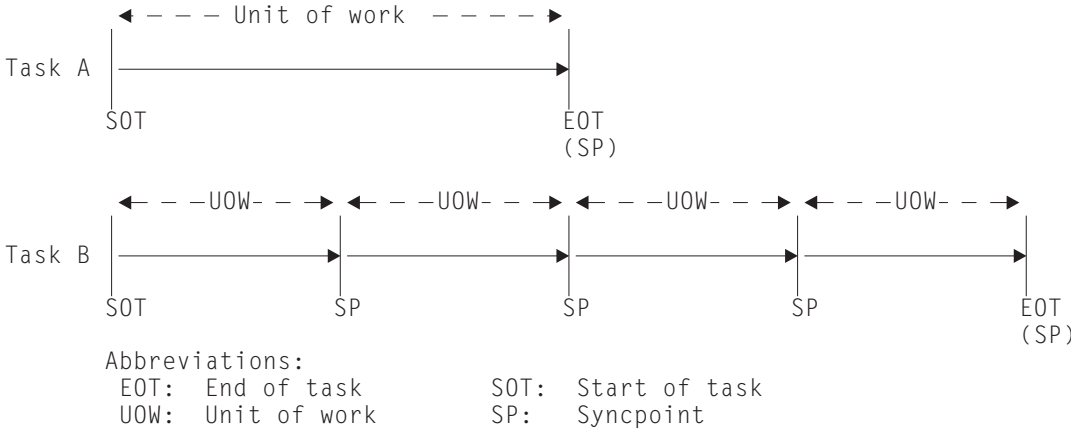



*Figure 13. Units of work and syncpoints*

Figure 14 on page 81 shows that database changes made by a task are not committed until a syncpoint is executed. If task processing is interrupted because of a failure of any kind, changes made within the abending UOW are automatically backed out.

If there is a system failure at time X:

- The change(s) made in task A have been committed and are therefore not backed out.
- In task B, the changes shown as Mod 1 and Mod 2 have been committed, but the change shown as Mod 3 is **not** committed and is backed out.
- All the changes made in task C are backed out.

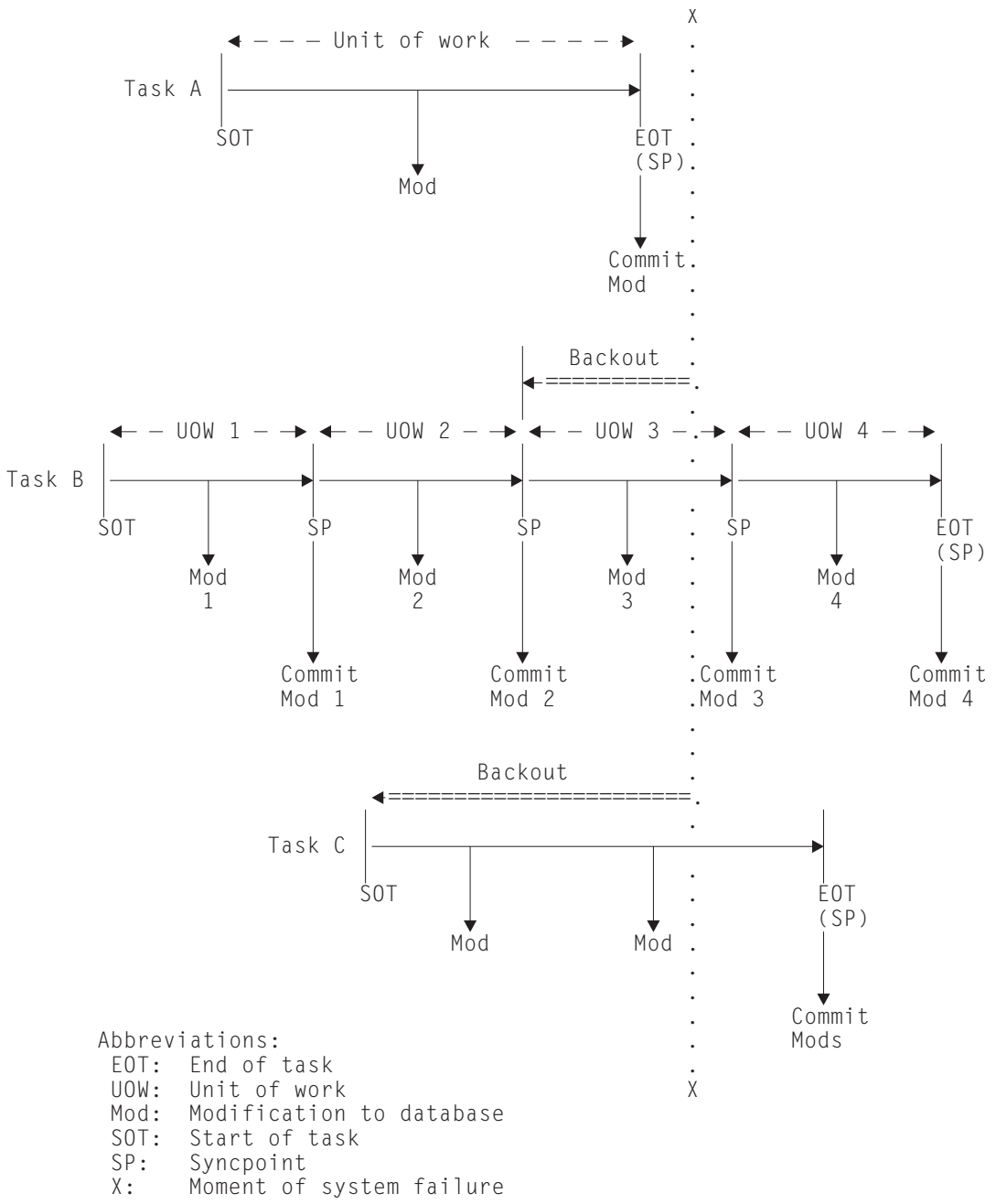



*Figure 14. Backout of units of work*

## CICS recovery manager

The recovery manager ensures the integrity and consistency of resources (such as files and databases) both within a single CICS region and distributed over interconnected systems in a network.

Figure 15 on page 82 shows the resource managers and their resources with which the CICS recovery manager works.

The main functions of the CICS recovery manager are:

- Managing the state, and controlling the execution, of each UOW
- Coordinating UOW-related changes during syncpoint processing for recoverable resources
- Coordinating UOW-related changes during restart processing for recoverable resources
- Coordinating recoverable conversations to remote nodes
- Temporarily suspending completion (**shunting**), and later resuming completion (**unshunting**), of UOWs that cannot immediately complete commit or backout processing because the required resources are unavailable, because of system, communication, or media failure

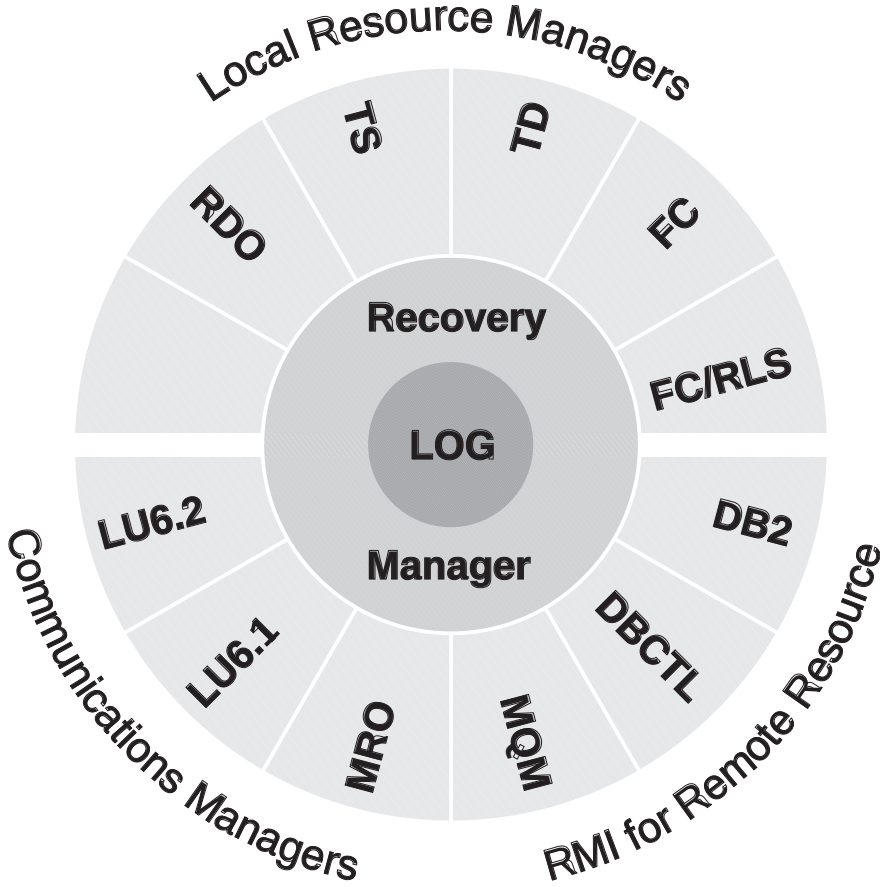



*Figure 15. CICS recovery manager and resources it works with*

## Managing the state of each unit of work

The CICS recovery manager maintains, for each UOW in a CICS region, a record of the changes of state that occur during its lifetime.

Typical events that cause state changes include:

- Creation of the UOW, with a unique identifier
- Premature termination of the UOW because of transaction failure
- Receipt of a syncpoint request
- Entry into the indoubt period during **two-phase commit** processing
- Notification that the resource is not available, requiring temporary suspension (shunting) of the UOW

- Notification that the resource is available, enabling retry of shunted UOWs
- Notification that a connection is reestablished, and can deliver a commit or rollback (backout) decision
- Syncpoint rollback
- Normal termination of the UOW

The identity of a UOW and its state are owned by the CICS recovery manager, and are recorded in storage and on the system log. The system log records are used by the CICS recovery manager during emergency restart to reconstruct the state of the UOWs in progress at the time of the earlier system failure.

The execution of a UOW can be distributed over more than one CICS system in a network of communicating systems.

The CICS recovery manager supports SPI commands that provide information about UOWs.

## Coordinating updates to local resources

The recoverable local resources managed by a CICS region are files, temporary storage, and transient data, plus resource definitions for terminals, typeterms, connections, and sessions.

Each local resource manager can write UOW-related log records to the local system log, which the CICS recovery manager may subsequently be required to re-present to the resource manager during recovery from failure.

To enable the CICS recovery manager to deliver log records to each resource manager as required, the CICS recovery manager adds additional information when the log records are created. Therefore, all logging by resource managers to the system log is performed through the CICS recovery manager.

During syncpoint processing, the CICS recovery manager invokes each local resource manager that has updated recoverable resources within the UOW. The local resource managers then perform the required action. This provides the means of coordinating the actions performed by individual resource managers.

If the commit or backout of a file resource fails (for example, because of an I/O error or the inability of a resource manager to free a lock), the CICS recovery manager takes appropriate action with regard to the failed resource:

- If the failure occurs during commit processing, the UOW is marked as commit-failed and is shunted awaiting resolution of whatever caused the commit failure.
- If the failure occurs during backout processing, the UOW is marked as backout-failed, and is shunted awaiting resolution of whatever caused the backout to fail.

Note that a commit failure can occur during the commit phase of a completed UOW, or the commit phase that takes place after successfully completing backout. (These two phases (or 'directions') of commit processing—commit after normal completion and commit after backout—are sometimes referred to as 'forward commit' and 'backward commit' respectively.) Note also that a UOW can be backout-failed with respect to some resources and commit-failed with respect to others. This can happen, for example, if two data sets are updated and the UOW has to be backed out, and the following happens:

- One resource backs out successfully
- While committing this successful backout, the commit fails
- The other resource fails to back out

These events leave one data set commit-failed, and the other backout-failed. In this situation, the overall status of the UOW is logged as backout-failed.

During emergency restart following a CICS failure, each UOW and its state is reconstructed from the system log. If any UOW is in the backout-failed or commit-failed state, CICS automatically retries the UOW to complete the backout or commit.

## Coordinating updates in distributed units of work

If the execution of a UOW is distributed across more than one system, the CICS recovery manager (or their non-CICS equivalents) in each pair of connected systems ensure that the effects of the distributed UOW are atomic.

Each CICS recovery manager (or its non-CICS equivalent) issues the requests necessary to effect two-phase syncpoint processing to each of the connected systems with which a UOW may be in conversation.

**Note:** In this context, the non-CICS equivalent of a CICS recovery manager could be the recovery component of a database manager, such as DBCTL or DB2, or any equivalent function where one of a pair of connected systems is not CICS.

In each connected system in a network, the CICS recovery manager uses interfaces to its local recovery manager connectors (RMCs) to communicate with partner recovery managers. The RMCs are the communication resource managers (IPIC, LU6.2, LU6.1, MRO, and RMI) which have the function of understanding the transport protocols and constructing the flows between the connected systems.

As remote resources are accessed during UOW execution, the CICS recovery manager keeps track of data describing the status of its end of the conversation with that RMC. The CICS recovery manager also assumes responsibility for the coordination of two-phase syncpoint processing for the RMC.

### Managing indoubt units of work

During the syncpoint phases, for each RMC, the CICS recovery manager records the changes in the status of the conversation, and also writes, on behalf of the RMC, equivalent information to the system log.

If a session fails at any time during the running of a UOW, it is the RMC responsibility to notify the CICS recovery manager, which takes appropriate action with regard to the unit of work as a whole. If the failure occurs during syncpoint processing, the CICS recovery manager may be in doubt and unable to determine immediately how to complete the UOW. In this case, the CICS recovery manager causes the UOW to be shunted awaiting UOW resolution, which follows notification from its RMC of successful resynchronization on the failed session.

During emergency restart following a CICS failure, each UOW and its state is reconstructed from the system log. If any UOW is in the indoubt state, it remains shunted awaiting resolution.

## Resynchronization after system or connection failure

Units of work that fail while in an indoubt state remain shunted until the indoubt state can be resolved following successful resynchronization with the coordinator.

Resynchronization takes place automatically when communications are next established between subordinate and coordinator. Any decisions held by the coordinator are passed to the subordinate, and indoubt units of work complete normally. If a subordinate has meanwhile taken a unilateral decision following the loss of communication, this decision is compared with that taken by the coordinator, and messages report any discrepancy.

For an explanation and illustration of the roles played by subordinate and coordinator CICS regions, and for information about recovery and resynchronization of distributed units of work generally, see Troubleshooting intersystem problems.

# CICS system log

CICS system log data is written to two MVS system logger log streams, the primary log stream and secondary log stream, which together form a single logical log stream.

The system log is the only place where CICS records information for use when backing out transactions, either dynamically or during emergency restart processing. CICS automatically connects to its system log stream during initialization, unless you have specified a journal model definition that defines the system log as DUMMY (in which case CICS can perform only an initial start).

The integrity of the system log is critical in enabling CICS to perform recovery. If any of the components involved with the system log—the CICS recovery manager, the CICS log manager, or the MVS system logger—experience problems with the system log, it might be impossible for CICS to perform successfully recovery processing. For more information about errors affecting the system log, see “Effect of problems with the system log” on page 96.

Setting up CICS log streams tells you more about CICS system log streams, and how you can use journal model definitions to map the CICS journal names for the primary system log stream (DFHLOG) and the secondary system log stream (DFHSHUNT) to specific log stream names. If you don't specify journal model definitions, CICS uses default log stream names.

## Information recorded on the system log

The information recorded on the system log is sufficient to allow backout of changes made to recoverable resources by transactions that were running at the time of failure, and to restore the recoverable part of CICS system tables.

Typically, this includes before-images of database records and after-images of recoverable parts of CICS tables—for example, transient data cursors or TCTTE sequence numbers. You cannot use the system log for forward recovery information, or for terminal control or file control autojournaling.

Your application programs can write user-defined recovery records to the system log using EXEC CICS WRITE JOURNALNAME commands. Any user-written log records to support your own recovery processes are made available to global user exit programs enabled at the XRCINPT exit point.

CICS also writes "backout-failed" records to the system log if a failure occurs in backout processing of a VSAM data set during dynamic backout or emergency restart backout.

Records on the system log are used for cold, warm, and emergency restarts of a CICS region. The only type of start for which the system log records are *not* used is an initial start.

### System activity keypoints

The recovery manager controls the recording of keypoint information, and the delivery of the information to the various resource managers at emergency restart.

The recovery manager provides the support that enables activity keypoint information to be recorded at frequent intervals on the system log. You specify the activity keypoint frequency on the **AKPFREQ** system initialization parameter. Activity keypoint information is of two types:

1. A list of all the UOWs currently active in the system
2. Image-copy type information allowing the current contents of a particular resource to be rebuilt

During an initial phase of CICS restart, recovery manager uses this information, together with UOW-related log records, to restore the CICS system to its state at the time of the previous shutdown. This is done on a single backward scan of the system log.

***Frequency of taking activity keypoints***: You are strongly recommended to specify a nonzero activity keypoint frequency. Choose an activity keypoint frequency that is suitable for the size of your system log stream. Note that writing activity keypoints at short intervals improves restart times, but at the expense of extra processing during normal running.

The following additional actions are taken for files accessed in non-RLS mode that use backup while open (BWO):

- Tie-up records are recorded on the forward recovery log stream. A tie-up record associates a CICS file name with a VSAM data set name.
- Recovery points are recorded in the integrated catalog facility (ICF) catalog. These define the starting time for forward recovery. Data recorded on the forward recovery log before that time does not need to be used.

## Forward recovery logs

CICS writes VSAM forward recovery logs to a general log stream defined to the MVS system logger. You can merge forward recovery log data for more than one VSAM data set to the same log stream, or you can dedicate a forward recovery log stream to a single data set.

See Defining forward recovery log streams for information about the use of forward recovery log streams.

## Replication logs

CICS writes VSAM replication logs to a general log stream defined to the MVS system logger. You can merge replication log data for more than one VSAM data set to the same log stream, or you can dedicate a replication log stream to a single data set.

See Replication logging for information about the use of replication logging.

## User journals and automatic journaling

User journals and autojournals are written to a general log stream defined to the MVS system logger.

- User journaling is entirely under your application programs' control. You write records for your own purpose using EXEC CICS WRITE JOURNALNAME commands. See “Flushing journal buffers” on page 90 for information about CICS shutdown considerations.
- Automatic journaling means that CICS automatically writes records to a log stream, referenced by the journal name specified in a journal model definition, as a result of:
  - Records read from or written to files. These records represent data that has been read, or data that has been read for update, or data that has been written, or records to indicate the completion of a write, and so on, depending on what types of request you selected for autojournaling.

    You specify that you want autojournaling for VSAM files using the autojournaling options on the file resource definition in the CSD. For BDAM files, you specify the options on a file entry in the file control table.
  - Input or output messages from terminals accessed through the z/OS Communications Server.

    You specify that you want terminal control autojournaling on the JOURNAL option of the profile resource definition referenced by your transaction definitions. These messages could be used to create audit trails.

  Automatic journaling is used for user-defined purposes; for example, for an audit trail. Automatic journaling is not used for CICS recovery purposes.

## Shutdown and restart recovery

CICS can shut down normally or abnormally and this affects the way that CICS restarts after it shuts down.

CICS can stop executing as a result of:

- A normal (warm) shutdown initiated by a CEMT, or EXEC CICS, PERFORM SHUT command
- An immediate shutdown initiated by a CEMT, or EXEC CICS, PERFORM SHUT IMMEDIATE command
- An abnormal shutdown caused by a CICS system module encountering an irrecoverable error
- An abnormal shutdown initiated by a request from the operating system (arising, for example, from a program check or system abend)
- A machine check or power failure

When CICS performs a WARM or EMERGENCY restart, resources that were installed during the previous run are reinstalled in exactly the same state. Avoid

changing system initialization parameters because this might cause initialization failures. For example, if you change a security system initialization parameter, Sockets Domain might fail to initialize because the required security mechanism cannot be found.

## Normal shutdown processing

Normal shutdown is initiated by issuing a CEMT PERFORM SHUTDOWN command, or by an application program issuing an EXEC CICS PERFORM SHUTDOWN command. It takes place in three quiesce stages, as follows:

### First quiesce stage

During the first quiesce stage of shutdown, all terminals are active, all CICS facilities are available, and the a number of activities are performed concurrently

The following activities are performed:

- CICS invokes the **shutdown assist transaction** specified on the SDTRAN system initialization parameter or on the shutdown command.

  Because all user tasks must terminate during the first quiesce stage, it is possible that shutdown could be unacceptably delayed by long-running tasks (such as conversational transactions). The purpose of the shutdown assist transaction is to allow as many tasks as possible to commit or back out cleanly, while ensuring that shutdown completes within a reasonable time.

  CICS obtains the name of the shutdown assist transaction as follows:

  1. If SDTRAN(*tranid*) is specified on the PERFORM SHUTDOWN command, or as a system initialization parameter, CICS invokes that *tranid*.
  2. If NOSDTRAN is specified on the PERFORM SHUTDOWN command, or as a system initialization parameter, CICS does not start a shutdown transaction. Without a shutdown assist transaction, all tasks that are already running are allowed to complete.
  3. If the SDTRAN (or NOSDTRAN) options are omitted from the PERFORM SHUTDOWN command, and omitted from the system initialization parameters, CICS invokes the default shutdown assist transaction, CESD, which runs the CICS-supplied program DFHCESD.

  The SDTRAN option specified on the PERFORM SHUT command overrides any SDTRAN option specified as a system initialization parameter.
- The DFHCESD program started by the CICS-supplied transaction, CESD, attempts to purge and back out long-running tasks using increasingly stronger methods (see "The shutdown assist transaction" on page 92).
- Tasks that are automatically initiated are run—if they start before the second quiesce stage.
- Any programs listed in the first part of the shutdown program list table (PLT) are run sequentially. (The shutdown PLT suffix is specified in the PLTSD system initialization parameter, which can be overridden by the PLT option of the CEMT or EXEC CICS PERFORM SHUTDOWN command.)
- A new task started as a result of terminal input is allowed to start only if its transaction code is listed in the current transaction list table (XLT) or has been defined as SHUTDOWN(ENABLED) in the transaction resource definition. The XLT list of transactions restricts the tasks that can be started by terminals and allows the system to shut down in a controlled manner. The current XLT is the one specified by the XLT=xx system initialization parameter, which can be overridden by the XLT option of the CEMT or EXEC CICS PERFORM SHUTDOWN command.

Certain CICS-supplied transactions are, however, allowed to start whether their code is listed in the XLT or not. These transactions are CEMT, CESF, CLR1, CLR2, CLQ2, CLS1, CLS2, CSAC, CSTE, and CSNE.

- Finally, at the end of this stage and before the second stage of shutdown, CICS unbinds all the z/OS Communications Server SNA LUs and devices.

The first quiesce stage is complete when the last of the programs listed in the first part of the shutdown PLT has executed **and** all user tasks are complete. If the CICS-supplied shutdown transaction CESD is used, this stage does not wait indefinitely for all user tasks to complete.

### Second quiesce stage

During the second quiesce stage of shutdown:

- Terminals are not active.
- No new tasks are allowed to start.
- Programs listed in the second part of the shutdown PLT (if any) run sequentially. These programs cannot communicate with terminals, or make any request that would cause a new task to start.

The second quiesce stage ends when the last of the programs listed in the PLT has completed executing.

### Third quiesce stage

During the third quiesce stage of shutdown:

- CICS closes all files that are defined to CICS file control. However, CICS does not catalog the files as UNENABLED; they can then be opened implicitly by the first reference after a subsequent CICS restart.

  Files that are eligible for BWO support have the BWO attributes in the ICF catalog set to indicate that BWO is not supported. This prevents BWO backups being taken in the subsequent batch window.
- All extrapartition TD queues are closed.
- CICS writes statistics to the system management facility (SMF) data set.
- CICS recovery manager sets the type-of-restart indicator in its domain state record in the global catalog to "warm-start-possible". If you specify START=AUTO when you next initialize the CICS region, CICS uses the status of this indicator to determine the type of startup it is to perform. See "How the state of the CICS region is reconstructed" on page 97.
- CICS writes warm keypoint records to:
  - The global catalog for terminal control and profiles
  - The CICS system log for all other resources.

  See "Warm keypoints" on page 90.
- CICS deletes all completed units of work (log tail deletion), leaving only shunted units of work and the warm keypoint.

  **Note:** Specifying no activity keypointing (AKPFREQ=0) only suppresses log tail deletion while CICS is running, not at shutdown. CICS always performs log clean up at shutdown unless you specify RETPD=dddd on the MVS definition of the system log. See Activity keypointing for more information.
- CICS stops executing.

## Warm keypoints

The CICS-provided warm keypoint program (DFHWKP) writes a warm keypoint to the global catalog, for terminal control and profile resources only, during the third quiesce stage of shutdown processing when all system activity is quiesced.

The remainder of the warm keypoint information, for all other resources, is written to the CICS system log stream, under the control of the CICS recovery manager. This system log warm keypoint is written by the activity keypoint program as a special form of activity keypoint that contains information relating to shutdown.

The warm keypoints contain information needed to restore the CICS environment during a subsequent warm or emergency restart. Thus CICS needs both the global catalog and the system log to perform a restart. If you run CICS with a system log that is defined by a journal model specifying TYPE(DUMMY), you cannot restart CICS with START=AUTO following a normal shutdown, or with START=COLD.

## Shunted units of work at shutdown

If there are shunted units of work of any kind at shutdown, CICS issues message DFHRM0203.

This message displays the numbers of indoubt, backout-failed, and commit-failed units of work held in the CICS region's system log at the time of the normal shutdown. It is issued only if there is at least one such UOW. If there are no shunted units of work, CICS issues message DFHRM0204.

DFHRM0203 is an important message that should be logged, and should be taken note of when you next restart the CICS region. For example, if you receive DFHRM0203 indicating that there is outstanding work waiting to be completed, you should not perform a cold or initial start of the CICS region. You are recommended to always restart CICS with START=AUTO, and especially after message DFHRM0203, otherwise recovery data is lost.

See “CICS cold start” on page 103 for information about a cold start if CICS has issued message DFHRM0203 at the previous shutdown.

## Flushing journal buffers

During a successful normal shutdown, CICS calls the log manager domain to flush all journal buffers, ensuring that all journal records are written to their corresponding MVS system logger log streams.

During an immediate shutdown, the call to the log manager domain is bypassed and journal records are not flushed. Therefore, any user journal records in a log manager buffer at the time of an immediate shutdown are lost. This does not affect CICS system data integrity. The system log and forward recovery logs are always synchronized with regard to I/O and unit of work activity. If user journal data is important, you should take appropriate steps to ensure that journal buffers are flushed at shutdown.

These situations and possible solutions are summarized as follows:

- In a controlled shutdown that completes normally, CICS ensures that user journals are flushed.
- Use a shutdown assist transaction. In a controlled shutdown that is forced into an immediate shutdown by a shutdown-assist transaction, the shutdown-assist transaction will flush the journal buffers before forcing an immediate shutdown.

- In an uncontrolled shutdown explicitly requested with the **SHUT IMMEDIATE** command, CICS does not flush buffers. To avoid the potential loss of journal records in this case, you can issue an **EXEC CICS WAIT JOURNALNAME** command at appropriate points in the application program, or immediately before returning control to CICS. (Alternatively, you could specify the WAIT option on the **WRITE JOURNALNAME** command.)

## Immediate shutdown processing (PERFORM SHUTDOWN IMMEDIATE)

As a general rule when terminating CICS, you are recommended to use a normal shutdown with a shutdown assist transaction, specifying either your own or the CICS-supplied default, CESD.

### PERFORM IMMEDIATE not recommended

You should resort to using an immediate shutdown only if you have a special reason for doing so. For instance, you might need to stop and restart CICS during a particularly busy period, when the slightly faster immediate shutdown may be of benefit. Also, you can use z/OS Communications Server persistent sessions support with an immediate shutdown.

You initiate an immediate shutdown by a CEMT, or EXEC CICS, PERFORM SHUTDOWN IMMEDIATE command. Immediate shutdown is different from a normal shutdown in a number of important ways:

1. If the shutdown assist transaction is not run (that is, the SDTRAN system initialization parameter specifies NO, or the PERFORM SHUTDOWN command specifies NOSDTRAN), user tasks are not guaranteed to complete. This can lead to an unacceptable number of units of work being shunted, with locks being retained.
2. If the default shutdown assist transaction CESD is run, it allows as many tasks as possible to commit or back out cleanly, but within a shorter time than that allowed on a normal shutdown. See “The shutdown assist transaction” on page 92 for more information about CESD, which runs the CICS-supplied program DFHCESD.
3. None of the programs listed in the shutdown PLT is executed.
4. CICS does **not** write a warm keypoint or a warm-start-possible indicator to the global catalog.
5. CICS does not close files managed by file control. It is left to VSAM to close the files when VSAM is notified by MVS that the address space is terminating. This form of closing files means that a VSAM VERIFY is needed on the next open of the files closed in this way, but this is done automatically.
6. Communications Server sessions wait for the restarted region to initialize or until the expiry of the interval specified in the PSDINT system initialization parameter, whichever is earlier.

The next initialization of CICS *must* be an emergency restart, in order to preserve data integrity. An emergency restart is ensured if the next initialization of CICS specifies START=AUTO. This is because the recovery manager's type-of-restart indicator is set to “emergency-restart-needed” during initialization and is not reset in the event of an immediate or uncontrolled shutdown. See “How the state of the CICS region is reconstructed” on page 97.

**Note:** A PERFORM SHUTDOWN IMMEDIATE command can be issued, by the operator or by the shutdown assist transaction, while a normal or immediate

shutdown is already in progress. If this happens, the shutdown assist transaction is not restarted; the effect is to force an immediate shutdown with no shutdown assist transaction.

If the original PERFORM SHUTDOWN request specified a normal shutdown, and the restart manager (ARM) was active, CICS is restarted (because CICS will not de-register from the automatic restart manager until the second quiesce stage of shutdown has completed).

## Shutdown requested by the operating system

This type of shutdown can be initiated by the operating system as a result of a program check or an operating system abend.

A program check or system abend can cause either an individual transaction to abend or CICS to terminate. (For further details, see Processing operating system abends and program checks.)

A CICS termination caused by an operating system request:

- Does not guarantee that user tasks will complete.
- Does not allow shutdown PLT programs to execute.
- Does **not** write a warm keypoint or a warm-start-possible indicator to the global catalog.
- Takes a system dump (unless system dumps are suppressed by the DUMP=NO system initialization parameter).
- Does not close any open files. It is left to VSAM to close the files when VSAM is notified by MVS that the address space is terminating. This form of closing files means that a VSAM VERIFY is needed on the next open of the files closed in this way, but this is done automatically.

The next initialization of CICS *must* be an emergency restart, in order to preserve data integrity. An emergency restart is ensured if the next initialization of CICS specifies START=AUTO. This is because the recovery manager's type-of-restart indicator is set to "emergency-restart-needed" during initialization, and is not reset in the event of an immediate or uncontrolled shutdown.

## Uncontrolled termination

An uncontrolled shutdown of CICS can be caused by a power failure, machine check, or operating system failure.

In each case, CICS cannot perform any shutdown processing. In particular, CICS does **not** write a warm keypoint or a warm-start-possible indicator to the global catalog.

The next initialization of CICS must be an emergency restart, in order to preserve data integrity. An emergency restart is ensured if the next initialization of CICS specifies START=AUTO. This is because the recovery manager's type-of-restart indicator is set to "emergency-restart-needed" during initialization, and is not reset in the event of an immediate or uncontrolled shutdown.

## The shutdown assist transaction

On an immediate shutdown, CICS does not allow running tasks to finish. A backout is not performed until an emergency restart.

This can cause an unacceptable number of units of work to be shunted, with locks being retained. On the other hand, on a normal shutdown, CICS waits indefinitely for running transactions to finish, which can delay shutdown to a degree that is unacceptable. The CICS shutdown assist transaction improves both these forms of shutdown and, to a large degree, removes the need for an immediate shutdown.

The operation of CESD, for both normal and immediate shutdowns, takes place over a number of stages. CESD controls these stages by sampling the number of tasks present in the system, and proceeds to the next stage if the number of in-flight tasks is not reducing quickly enough.

The stages of a normal shutdown CESD are as follows:

- In the initial stage of assisted shutdown, CESD attempts to complete a normal shutdown in a reasonable time.
- After a time allowed for transactions to finish normally (that is, after the number of tasks has not reduced over a period of eight samples), CESD proceeds to issue a normal purge for each remaining task. The transaction dump data set is closed in this stage.
- If there are still transactions running after a further eight samples (except when persistent sessions support is being used), the z/Os Communications Server is force-closed and IRC is closed immediately.
- Finally, if there are still transactions running, CICS shuts down abnormally, leaving details of the remaining in-flight transactions on the system log to be dealt with during an emergency restart.

The operation of CESD is quicker for an immediate shutdown, with the number of tasks in the system being sampled only four times instead of eight.

You are recommended always to use the CESD shutdown-assist transaction when shutting down your CICS regions. You can use the DFHCESD program "as is", or use the supplied source code as the basis for your own customized version (CICS supplies versions in assembler, COBOL, and PL/I).

## Cataloging CICS resources

CICS uses a global catalog data set (DFHGCD) and a local catalog data set (DFHLCD) to store information that is passed from one execution of CICS, through a shutdown, to the next execution of CICS.

This information is used for warm and emergency restarts, and to a lesser extent for cold starts. If the global catalog fails (for reasons other than filling the available space), the recovery manager control record is lost. Without this, it is impossible to perform a warm, emergency, or cold start, and the only possibility is then an initial start. For example, if the failure is due to an I/O error, you cannot restart CICS.

Usually, if the global catalog fills, CICS abnormally terminates, in which case you could define more space and attempt an emergency restart.

Consider putting the catalog data sets on the most reliable storage available—RAID or dual-copy devices—to ensure maximum protection of the data. Taking ordinary copies is not recommended because of the risk of getting out of step with the system log.

From a restart point of view, the system log and the CICS catalog (both data sets) form one logical set of data, and all of them are required for a restart.

Setting up the catalog data sets tells you how to create and initialize these CICS catalog data sets.

## Global catalog

The global catalog contains information that CICS requires on a restart.

CICS uses the global catalog to store the following information:

- The names of the system log streams.
- Copies of tables of installed resource definitions, and related information, for the following:
  - Transactions and transaction classes
  - DB2 resource definitions
  - Programs, mapsets, and partitionsets (including autoinstalled programs, subject to the operand you specify on the PGAICTLG system initialization parameter)
  - Terminals and typeterms (for predefined and autoinstalled resources)
  - Autoinstall terminal models
  - Profiles
  - Connections, sessions, and partners
  - BDAM and VSAM files (including data tables) and
    - VSAM LSR pool share control blocks
    - Data set names and data set name blocks
    - File control recovery blocks (only if a SHCDS NONRLSUPDATEPERMITTED command has been used).
  - Transient data queue definitions
  - Dump table information
  - Interval control elements and automatic initiate descriptors at shutdown
  - APPC connection information so that relevant values can be restored during a persistent sessions restart
  - Logname information used for communications resynchronization
  - Monitoring options in force at shutdown
  - Statistics interval collection options in force at shutdown
  - Journal model and journal name definitions
  - Enqueue model definitions
  - Temporary storage model definitions
  - URIMAP definitions and virtual hosts for CICS Web support.

  Most resource managers update the catalog whenever they make a change to their table entries. Terminal and profile resource definitions are exceptions (see the next list item about the catalog warm keypoint). Because of the typical volume of changes, terminal control does not update the catalog, except when:
  - Running a z/OS Communications Server query against a terminal
  - A generic connection has bound to a remote system
  - Installing a terminal
  - Deleting a terminal.
- A partial warm keypoint at normal shutdown. This keypoint contains an image copy of the TCT and profile resource definitions at shutdown for use during a warm restart.

**Note:** The image copy of the TCT includes all the permanent devices installed by explicit resource definitions. Except for some autoinstalled APPC connections, it does *not* include autoinstalled devices. Autoinstalled terminal resources are cataloged initially, in case they need to be recovered during an emergency restart, but only if the AIRDELAY system initialization parameter specifies a nonzero value. Therefore, apart from the APPC exceptions previously mentioned, autoinstalled devices are excluded from the warm keypoint, and are thus not recovered on a warm start.

- Statistics options.
- Monitoring options.
- The recovery manager's control record, which includes the type-of-restart indicator (see "How the state of the CICS region is reconstructed" on page 97).

All this information is essential for a successful restart following any kind of shutdown.

### Local catalog

The CICS local catalog data set represents just one part of the CICS catalog, which is implemented as two physical data sets.

The two data sets are logically one set of cataloged data managed by the CICS catalog domain. Although minor in terms of the volume of information recorded on it, the local catalog is of equal importance with the global catalog, and the data should be equally protected when restarts are performed.

If you ever need to redefine and reinitialize the CICS local catalog, you should also reinitialize the global catalog. After reinitializing both catalog data sets, you must perform an initial start.

## Shutdown initiated by CICS log manager

The CICS log manager initiates a shutdown of the region if it encounters an error in the system log that indicates previously logged data has been lost.

In addition to initiating the shutdown, the log manager informs the recovery manager of the failure, which causes the recovery manager to set the type-of-restart indicator to "no-restart-possible" and to issue message DFHRM0144. The result is that recovery during a subsequent restart is not possible and you can perform only an initial start of the region. To do this you are recommended to run the recovery manager utility program (DFHRMUTL) to force an initial start, using the SET_AUTO_START=AUTOINIT option.

During shutdown processing, existing transactions are given the chance to complete their processing. However, no further data is written to the system log. This strategy ensures that the minimum number of units of work are impacted by the failure of the system log. This is because:

- If a unit of work does not attempt to backout its resource updates, and completes successfully, it is unaffected by the failure.
- If a unit of work does attempt to backout, it cannot rely on the necessary log records being available, and therefore it is permanently suspended.

Therefore, when the system has completed the log manager-initiated shutdown all (or most) units of work will have completed normally during this period and if there are no backout attempts, data integrity is not compromised.

## Effect of problems with the system log

A key value of CICS is its ability to implement its transactional recovery commitments and thus safeguard the integrity of recoverable data updated by CICS applications.

This ability relies upon logging before-images and other information to the system log. However, the system log itself might suffer software or hardware related problems, including failures in the CICS recovery manager, the CICS logger domain, or the MVS system logger. Although problems with these components are unlikely, you must understand the actions to take to minimize the impact of such problems.

If the CICS log manager detects an error in the system log that indicates previously logged data has been lost, it initiates a shutdown of the region. This action minimizes the number of transactions that fail after a problem with the log is detected and therefore minimizes the data integrity exposure.

Any problem with the system log that indicates that it might not be able to access all the data previously logged invalidates the log. In this case, you can perform only a diagnostic run or an initial start of the region to which the system log belongs.

The reason that a system log is completely invalidated by these kinds of error is that CICS can no longer rely on the data it previously logged being available for recovery processing. For example, the last records logged might be unavailable, and therefore recovery of the most recent units of work cannot be carried out. However, data might be missing from any part of the system log and CICS cannot identify what is missing. CICS cannot examine the log and determine exactly what data is missing, because the log data might appear consistent in itself even when CICS has detected that some data is missing.

These are the messages that CICS issues as it reads the log during system initialization except when START=INITIAL is specified. They can help you identify which units of work were recovered:

**DFHRM0402**
: This message is issued for each unit of work when it is first encountered on the log.

**DFHRM0403 and DFHRM0404**
: One of these messages is issued for each unit of work when its context is found. The message reports the state of the unit of work.

**DFHRM0405**
: This message is issued when a complete keypoint has been recovered from the log.

If you see that message DFHRM0402 is issued for a unit of work, and it is matched by message DFHRM0403 or DFHRM0404, you can be sure of the state of the unit of work. If you see message DFHRM0405, you can use the preceding messages to determine which units of work are incomplete, and you can also be sure that none of the units of work is completely missing.

Another class of problem with the system log is one that does not indicate any loss of previously logged data; for example, access to the logstream was lost due to termination of the MVS system logger address space. This class of problem causes

an immediate termination of CICS because a subsequent emergency restart will probably succeed when the cause of the problem has been resolved.

For information about how to deal with system log problems, see Some conditions that cause CICS log manager error messages.

## How the state of the CICS region is reconstructed

CICS recovery manager uses the type-of-restart indicator in its domain state record from the global catalog to determine which type of restart it is to perform.

This indicator operates as follows:

- Before the end of initialization, on all types of startup, CICS sets the indicator in the control record to “emergency restart needed”.
- If CICS terminates normally, this indicator is changed to “warm start possible”.
- If CICS terminates abnormally because the system log has been corrupted and is no longer usable, this indicator is changed to “no restart”. After fixing the system log, perform an initial start of the failed CICS region.
- For an automatic start (START=AUTO):
  - If the indicator says “warm start possible”, CICS performs a warm start.
  - If the indicator says “emergency restart needed”, CICS performs an emergency restart.

### Overriding the type of start indicator

The operation of the recovery manager's control record can be modified by running the recovery manager utility program, DFHRMUTL.

#### About this task

This can set an autostart record that determines the type of start CICS is to perform, effectively overriding the type of start indicator in the control record. See Recovery manager utility (DFHRMUTL) for information about using DFHRMUTL to modify the type of start performed by START=AUTO.

### Warm restart

If you shut down a CICS region normally, CICS restarts with a warm restart if you specify START=AUTO. For a warm start to succeed, CICS needs the information stored in the CICS catalogs at the previous shutdown, and the information stored in the system log.

In a warm restart, CICS:

1. Restores the state of the CICS region to the state it was in at completion of the normal shutdown. All CICS resource definitions are restored from the global catalog, and the **GRPLIST**, **FCT**, and **CSD** system initialization parameters are ignored.

   CICS also uses information from the warm keypoint in the system log.
2. Reconnects to the system log.
3. Retries any backout-failed and commit-failed units of work.
4. Rebuilds indoubt-failed units of work.

For more information about the warm restart process, see “CICS warm restart” on page 110.

## Emergency restart

If a CICS region fails, CICS restarts with an emergency restart if you specify START=AUTO. An emergency restart is similar to a warm start but with additional recovery processing for example, to back out any transactions that were in-flight at the time of failure, and thus free any locks protecting resources.

If the failed CICS region was running with VSAM record-level sharing, SMSVSAM converts into retained locks any active exclusive locks held by the failed system, pending the CICS restart. This means that the records are protected from being updated by any other CICS region in the sysplex. Retained locks also ensure that other regions trying to access the protected records do not wait on the locks until the failed region restarts. See Exclusive locks and shared locks for information about active and retained locks.

For non-RLS data sets (including BDAM data sets), any locks (ENQUEUES) that were held before the CICS failure are reacquired.

### Initialization during emergency restart

Most of CICS initialization following an emergency restart is the same as for a warm restart, and CICS uses the catalogs and the system log to restore the state of the CICS region. Then, after the normal initialization process, emergency restart performs the recovery process for work that was in-flight when the previous run of CICS was abnormally terminated.

### Recovery of data during an emergency restart

During the final stage of emergency restart, the recovery manager uses the system log data to drive backout processing for any units of work that were in-flight at the time of the failure. The backout of units of work during emergency restart is the same as a dynamic backout; there is no distinction between the backout that takes place at emergency restart and that which takes place at any other time.

The recovery manager also drives:

- The backout processing for any units of work that were in a backout-failed state at the time of the CICS failure.
- The commit processing for any units of work that were in a commit-failed state at the time of the CICS failure.
- The commit processing for units of work that had not completed commit at the time of failure (resource definition recovery, for example).

The recovery manager drives these backout and commit processes because the condition that caused them to fail may be resolved by the time CICS restarts. If the condition that caused a failure has not been resolved, the unit of work remains in backout- or commit-failed state. See Backout-failed recovery and Commit-failed recovery for more information.

For more information about the emergency restart process, see CICS emergency restart.

## Cold start

On a cold start, CICS reconstructs the state of the region from the previous run for remote resources only. For all resources, the region is built from resource definitions specified on the **GRPLIST** system initialization parameter and those resources defined in control tables.

The following is a summary of how CICS uses information stored in the global catalog and the system log on a cold start:

- CICS preserves, in both the global catalog and the system log, all the information relating to distributed units of work for partners linked by:
  - APPC
  - MRO connections to regions running under CICS Transaction Server
  - The resource manager interface (RMI); for example, to DB2 and DBCTL.
- CICS does not preserve any information in the global catalog or the system log that relates to local units of work.

Generally, to perform a cold start you specify START=COLD, but CICS can also force a cold start in some circumstances when START=AUTO is specified. See START parameter for details of the effect of the **START** parameter in conjunction with various states of the global catalog and the system log.

**An initial start of CICS:**

If you want to initialize a CICS region without reference to the global catalog from a previous run, perform an initial start.

You can do this by specifying START=INITIAL as a system initialization parameter, or by running the recovery manager's utility program (DFHRMUTL) to override the type of start indicator to force an initial start.

See Recovery manager utility (DFHRMUTL) for information about the DFHRMUTL utility program.

## Dynamic RLS restart

If a CICS region is connected to an SMSVSAM server when the server fails, CICS continues running, and recovers using a process known as dynamic RLS restart. An SMSVSAM server failure does not cause CICS to fail, and does not affect any resource other than data sets opened in RLS mode.

When an SMSVSAM server fails, any locks for which it was responsible are converted to retained locks by another SMSVSAM server within the sysplex, thus preventing access to the records until the situation has been recovered. CICS detects that the SMSVSAM server has failed the next time it tries to perform an RLS access after the failure, and issues message DFHFC0153. The CICS regions that were using the failed SMSVSAM server defer in-flight transactions by abending units of work that attempt to access RLS, and shunt them when the backouts fail with “RLS is disabled” responses. If a unit of work is attempting to commit its changes and release RLS locks, commit failure processing is invoked when CICS first detects that the SMSVSAM server is not available (see Commit-failed recovery).

RLS mode open requests and RLS mode record access requests issued by new units of work receive error responses from VSAM when the server has failed. The SMSVSAM server normally restarts itself without any manual intervention. After the SMSVSAM server has restarted, it uses the MVS event notification facility (ENF) to notify all the CICS regions within its MVS image that the SMSVSAM server is available again.

CICS performs a dynamic equivalent of emergency restart for the RLS component, and drives backout of the deferred work.

Recovery after the failure of an SMSVSAM server is usually performed automatically by CICS. CICS retries any backout-failed and commit-failed units of work. In addition to retrying those failed as a result of the SMSVSAM server failure, this also provides an opportunity to retry any backout failures for which the cause has now been resolved. Manual intervention is required only if there are units of work which, due to the timing of their failure, were not retried when CICS received the ENF signal. This situation is extremely unlikely, and such units of work can be detected using the **INQUIRE UOWDSNFAIL** command.

Note that an SMSVSAM server failure causes commit-failed or backout-failed units of work only in the CICS regions registered with the server in the same MVS image. Transactions running in CICS regions in other MVS images within the sysplex are affected only to the extent that they receive LOCKED responses if they try to access records protected by retained locks owned by any CICS regions that were using the failed SMSVSAM server.

## Recovery with z/OS Communications Server persistent sessions

With z/OS Communications Server persistent sessions support, if CICS fails or undergoes immediate shutdown (by means of a **PERFORM SHUTDOWN IMMEDIATE** command), the Communications Server holds the CICS LU-LU sessions in recovery-pending state, and they can be recovered during startup by a newly starting CICS region. With multinode persistent sessions support, sessions can also be recovered if the Communications Server or z/OS fails in a sysplex.

The CICS system initialization parameter **PSTYPE** specifies the type of persistent sessions support for a CICS region:

**SNPS, single-node persistent sessions**
: Persistent sessions support is available, so that Communications Server sessions can be recovered after a CICS failure and restart. This setting is the default.

**MNPS, multinode persistent sessions**
: In addition to the SNPS support, Communications Server sessions can also be recovered after a Communications Server or z/OS failure in a sysplex.

**NOPS, no persistent sessions**
: Persistent sessions support is not required for the CICS region. For example, a CICS region that is used only for development or testing might not require persistent sessions.

For single-node persistent sessions support, you require z/OS Communications Server V3.4.1 or later, which supports persistent LU-LU sessions. CICS Transaction Server for z/OS, Version 5 Release 4 functions with releases of z/OS Communications Server earlier than V3.4.1, but in the earlier releases sessions are not retained in a bound state if CICS fails. For multinode persistent sessions support, you require z/OS Communications Server V4.R4 or later, and z/OS Communications Server must be in a Parallel Sysplex® with a coupling facility.

CICS support of persistent sessions includes the support of all LU-LU sessions, except LU0 pipeline and LU6.1 sessions. With multinode persistent sessions support, if the Communications Server or z/OS fails, LU62 synclevel 1 sessions are restored, but LU62 synclevel 2 sessions are not restored.

## Running with persistent sessions support

When you specify SNPS or MNPS for the **PSTYPE** system initialization parameter so that z/OS Communications Server persistent sessions support is in use for a CICS region, the time specified by the **PSDINT** system initialization parameter for the region determines how long the sessions are retained.

If a CICS, Communications Server, or z/OS failure occurs, if a connection to the Communications Server is reestablished within this time, CICS can use the retained sessions immediately; there is no need for network flows to rebind them.

Make sure that you set a nonzero value for the persistent sessions delay interval, so that sessions are retained. The default is zero, which means that persistent sessions support is available if you have specified SNPS or MNPS for **PSTYPE**, but it is not being exploited.

You can change the persistent sessions delay interval using the **CEMT SET VTAM** command, or the **EXEC CICS SET VTAM** command. The changed interval is not stored in the CICS global catalog, and therefore is not restored in an emergency restart.

**Note:** VTAM is now the z/OS Communications Server.

During an emergency restart of CICS, CICS restores those sessions pending recovery from the CICS global catalog and the CICS system log to an in-session state. This process of persistent sessions recovery takes place when CICS opens its VTAM ACB. With multinode persistent sessions support, if the Communications Server or z/OS fails, sessions are restored when CICS reopens its VTAM ACB, either automatically by the COVR transaction, or by a CEMT or **EXEC CICS SET VTAM OPEN** command. Although sessions are recovered, any transactions inflight at the time of the failure are abended and not recovered.

When a terminal user enters data during persistent sessions recovery, CICS appears to hang. The screen that was displayed at the time of the failure remains on display until persistent sessions recovery is complete. You can use options on the TYPETERM and SESSIONS resource definitions for the CICS region to customize CICS so that either a successful recovery can be transparent to terminal users, or terminal users can be notified of the recovery, allowing them to take the appropriate actions.

If APPC sessions are active at the time of the CICS, Communications Server or z/OS failure, persistent sessions recovery appears to APPC partners as CICS hanging. The Communications Server saves requests issued by the APPC partner, and passes them to CICS when recovery is complete. When CICS reestablishes a connection with the Communications Server, recovery of terminal sessions is determined by the settings for the PSRECOVERY option of the CONNECTION resource definition and the RECOVOPTION option of the SESSIONS resource definition. You must set the PSRECOVERY option of the CONNECTION resource definition to the default value SYSDEFAULT for sessions to be recovered. The alternative, NONE, means that no sessions are recovered. If you have selected the appropriate recovery options and the APPC sessions are in the correct state, CICS performs an **ISSUE ABEND** to inform the partner that the current conversation has been abnormally ended.

If CICS has persistent verification defined, the sign-on is not active under persistent sessions until the first input is received by CICS from the terminal.

Defining z/OS Communications Server persistent sessions support describes the steps required to define persistent sessions support for a CICS region.

**Situations in which sessions are not reestablished:**

When z/OS Communications Server persistent sessions support is in use for a CICS region, CICS does not always reestablish sessions that are being held by the Communications Server in a recovery pending state. In the situations listed here, CICS or the Communications Server unbinds and does not rebind recovery pending sessions.

- If CICS does not restart within the persistent sessions delay interval, as specified by the **PSDINT** system initialization parameter.
- If you perform a COLD start after a CICS failure.
- If CICS cannot find a terminal control table terminal entry (TCTTE) for a session; for example, because the terminal was autoinstalled with AIRDELAY=0 specified.
- If a terminal or session is defined with the recovery option (RECOVOPTION) of the TYPETERM or SESSIONS resource definition set to RELEASESESS, UNCONDREL or NONE.
- If a connection is defined with the persistent sessions recovery option (PSRECOVERY) of the CONNECTION resource definition set to NONE.
- If CICS determines that it cannot recover the session without unbinding and rebinding it.

The result in each case is as if CICS has restarted following a failure without Communications Server persistent sessions support.

In some other situations APPC sessions are unbound. For example, if a bind was in progress at the time of the failure, sessions are unbound.

With multinode persistent sessions support, if a Communications Server or z/OS failure occurs and the TPEND failure exit is driven, the autoinstalled terminals that are normally deleted at this point are retained by CICS. If the session is not reestablished and the terminal is not reused within the AIRDELAY interval, CICS deletes the TCTTE when the AIRDELAY interval expires after the ACB is reopened successfully.

**Situations in which the z/OS Communications Server does not retain sessions:**

When z/OS Communications Server persistent sessions support is in use for a CICS region, in some circumstances the Communications Server does not retain LU-LU sessions.

- If you close the Communications Server with any of the following CICS commands:
  - **SET VTAM FORCECLOSE**
  - **SET VTAM IMMCLOSE**
  - **SET VTAM CLOSED**

  **Note:** VTAM is now the z/OS Communications Server.
- If you close the CICS node with the Communications Server command **VARY NET INACT ID**=*applid*.
- If your CICS system performs a normal shutdown, with a **PERFORM SHUTDOWN** command.

If single-node persistent sessions support (SNPS), which is the default, is specified for a CICS region, sessions are not retained after a Communications Server or z/OS failure. If multinode persistent sessions support (MNPS) is specified, sessions are retained after a Communications Server or z/OS failure.

### Running without persistent sessions support

z/OS Communications Server persistent sessions support is the default for a CICS region, but you might choose to run a CICS region without this support if it is used only for development or testing. Specify NOPS for the **PSTYPE** system initialization parameter to start a CICS region without persistent sessions support. Running without persistent sessions support can enable you to increase the number of CICS regions in an LPAR.

If you have a large number of CICS regions in the same LPAR (around 500), with persistent sessions support available for all the regions, you might reach a z/OS limit on the maximum number of data spaces and be unable to add any more CICS regions. In this situation, when you attempt to start further CICS regions, you see messages IST967I and DFHSI1572, stating that the ALESERV ADD request has failed and the VTAM (z/OS Communications Server) ACB cannot be opened. However, a region without persistent sessions support does not use a data space and so does not count towards the limit. To obtain a greater number of CICS regions in the LPAR:

1. Identify existing regions that can run without persistent sessions support.
2. Change the **PSTYPE** system initialization parameter for those regions to specify NOPS, and specify a zero value for the **PSDINT** system initialization parameter.
3. Cold start the regions to implement the change.

You can then start further CICS regions with or without persistent sessions support as appropriate, provided that you do not exceed the limit for the number of regions that do have persistent sessions support.

If you specify NOPS (no persistent session support) for the **PSTYPE** system initialization parameter, a zero value is required for the **PSDINT** (persistent session delay interval) system initialization parameter.

When persistent sessions support is not in use, all sessions existing on a CICS system are lost if that CICS system, Communications Server, or z/OS fails. In any subsequent restart of CICS, the rebinding of sessions that existed before the failure depends on the AUTOCONNECT option for the terminal. If AUTOCONNECT is specified for a terminal, the user of that terminal waits until the GMTRAN transaction has run before being able to continue working. The user sees the Communications Server logon panel followed by the “good morning” message. If AUTOCONNECT is not specified for a terminal, the user of that terminal has no way of knowing (unless told by support staff) when CICS is operational again unless the user tries to log on. In either case, users are disconnected from CICS and need to reestablish a session, or sessions, to regain their working environment.

## CICS cold start

This section describes the CICS startup processing specific to a cold start.

It covers the two forms of cold start:


- “Starting CICS with the START=COLD parameter” on page 104
- “Starting CICS with the START=INITIAL parameter” on page 109

## Starting CICS with the START=COLD parameter

START=COLD performs a dual type of startup, performing a cold start for all local resources while preserving recovery information that relates to remote systems or resource managers connected through the resource manager interface (RMI).

This ensures the integrity of the CICS region with its partners in a network that manages a distributed workload. You can use a cold start to install resource definitions from the CSD (and from macro control tables). It is normally safe to perform a cold start for a CICS region that does not own any local resources (such as a terminal-owning region that performs only transaction routing). For more information about performing a cold start, and when it is safe to do so, see Initial and cold starts .

If you specify `START=COLD`, CICS either discards or preserves information in the system log and global catalog data set, as follows:

- CICS deletes all cataloged resource definitions in the CICS catalogs and installs definitions either from the CSD or from macro control tables. CICS writes a record of each definition in the global catalog data set as each resource definition is installed. All transaction and transaction class resource definitions, journal model definitions, programs, mapsets, and partitionsets are installed from the CSD, and are cataloged in the global catalog. Journal name definitions (including the system logs DFHLOG and DFHSHUNT) are created using the installed journal models and cataloged in the global catalog.
- Any program LIBRARY definitions that had been dynamically defined will be lost. Only the static DFHRPL concatenation will remain, together with any LIBRARY definitions in the grouplist specified at startup or installed via BAS at startup.
- CICS preserves the recovery manager control record, which contains the CICS logname token used in the previous run. CICS also preserves the log stream name of the system log.
- CICS discards any information from the system log that relates to *local* resources, and resets the system log to begin writing at the start of the primary log stream.

  **Note:** If CICS detects that there were shunted units of work at the previous shutdown (that is, it had issued message DFHRM0203) CICS issues a warning message, DFHRM0154, to let you know that local recovery data has been lost, and initialization continues. The only way to avoid this loss of data from the system log is *not* to perform a cold start after CICS has issued DFHRM0203.

  If the cold start is being performed following a shutdown that issued message DFHRM0204, CICS issues message DFHRM0156 to confirm that the cold start has not caused any loss of local recovery data.
- CICS releases *all* retained locks:
  - CICS requests the SMSVSAM server, if connected, to release all RLS retained locks.
  - CICS does not rebuild the non-RLS retained locks.
- CICS requests the SMSVSAM server to clear the RLS sharing control status for the region.
- CICS does not restore the dump table, which may contain entries controlling system and transaction dumps.
- CICS preserves resynchronization information about distributed units of work—information regarding unit of work obligations to remote systems, or to non-CICS resource managers (such as DB2) connected through the RMI. For

example, the preserved information includes data about the outcome of distributed UOWs that is needed to allow remote systems (or RMI resource managers) to resynchronize their resources.

**Note:** The system log information preserved does *not* include before-images of any file control data updated by a distributed unit of work. Any changes made to local file resources are not backed out, and by freeing all locks they are effectively committed. To preserve data integrity, perform a warm or emergency restart using START=AUTO.

- CICS retrieves its logname token from the recovery manager control record for use in the "exchange lognames" process during reconnection to partner systems. Thus, by using the logname token from the previous execution, CICS ensures a warm start of those connections for which there is outstanding resynchronization work.

To perform these actions on a cold start, CICS needs the contents of the catalog data sets and the system log from a previous run. The CICS log manager retrieves the system log stream name from the global catalog ensuring CICS uses the same log stream as on a previous run.

See for details of the actions that CICS takes for START=COLD in conjunction with various states of the global catalog and the system log.

The returns information about the type of previous CICS shutdown which is of use in determining whether a cold restart is possible or not.

The following information provides more detail about what happens to different CICS resources when you perform a cold start. This information applies to resources stored in the CSD and macro control tables. Resources that are defined in CICS bundles are not stored in the CSD, and they are recovered using different processes, depending on whether the CICS bundles were created from a definition or by platforms and applications. For details of how CICS handles BUNDLE resources at startup, see .

## CICS resources discarded when START=COLD is specfied

CICS will discard certain information in the system log and global catalog data set if you specify **START=COLD**.

### Files

All previous file control state data, including file resource definitions, is lost. See Files for more information.

### Temporary Storage

All temporary storage queues from a previous run are lost, including CICS-generated queues (for example, for data passed on START requests). If the auxiliary temporary storage data set was used on a previous run, CICS opens the data set for update. If CICS finds that the data set is newly initialized, CICS closes it, reopens it in output mode, and formats all the control intervals (CIs) in the primary extent. When formatting is complete, CICS closes the data set and reopens it in update mode. The time taken for this formatting operation depends on the size of the primary extent, but it can add significantly to the time taken to perform a cold start.

### Temporary storage data sharing server

Any queues written to a shared temporary storage pool normally persist across a cold start.

Shared TS pools are managed by a temporary storage server, and stored in the coupling facility. Stopping and restarting a TS data sharing server does not affect the contents of the TS pool, unless you clear the coupling facility structure in which the pool resides.

If you want to cause a server to reinitialize its pool, use the MVS SETXCF FORCE command to clean up the structure:

```
SETXCF FORCE,STRUCTURE,STRNAME(DFHXQLS_poolname)
```

The next time you start up the TS server following a SETXCF FORCE command, the server initializes its TS pool in the structure using the server startup parameters specified in the DFHXQMN job.

### Transient data

All transient data queues from a previous run are lost. Transient data resource definitions are installed from Resource groups defined in the CSD, as specified in the CSD group list (named on the GRPLIST system initialization parameter). Any extrapartition TD queues that require opening are opened; that is, any that specify OPEN(INITIAL). All the newly-installed TD queue definitions are written to the global catalog. All TD queues are installed as enabled.CSD definitions are installed later than the macro-defined entries because of the position of CSD group list processing in the initialization process. Any extrapartition TD queues that need to be opened are opened; that is, any that specify OPEN=INITIAL. The TDINTRA system initialization parameter has no effect in a cold start.

### LIBRARY resources

All LIBRARY resources from a previous run are lost. LIBRARY resource definitions are installed from resource groups defined in the CSD, as specified in the CSD group list (named on the GRPLIST system initialization parameter).

### Start requests (with and without a terminal)

All forms of start request recorded in a warm keypoint (if the previous shutdown was normal) are lost. This applies both to START requests issued by a user application program and to START commands issued internally by CICS in support of basic mapping support (BMS) paging. Any data associated with START requests is also lost, even if it was stored in a recoverable TS queue.

### Resource definitions dynamically installed

Any resource definitions dynamically added to a previous run of CICS are lost in a cold start, unless they are included in the group list specified on the GRPLIST system initialization parameter. If you define new resource definitions and install them dynamically, ensure the group containing the resources is added to the appropriate group list.

### Terminal control resources

All previous terminal control information stored in the global catalog warm keypoint is lost. See Terminal control resources for more information.

### Dump Table

The dump table that you use for controlling system and transaction dumps is not preserved in a cold start. If you have built up over a period of time a number of entries in a dump table, which is recorded in the CICS catalog, you have to re-create these entries following a cold start.

**Files:**

All previous file control state data, including file resource definitions, is lost.

If RLS support is specified, CICS connects to the SMSVSAM, and when connected requests the server to:

- Release all RLS retained locks
- Clear any "lost locks" status
- Clear any data sets in "non-RLS update permitted" status

For non-RLS files, the CICS enqueue domain does not rebuild the retained locks relating to shunted units of work.

File resource definitions are installed as follows:

**VSAM**
: Except for the CSD itself, all VSAM file definitions are installed from the CSD. You specify these in groups named in the CSD group list, which you specify on the GRPLIST system initialization parameter. The CSD file definition is built and installed from the CSD*xxxx* system initialization parameters.

**Data tables**
: As for VSAM file definitions.

**BDAM**
: File definitions are installed from file control table entries, specified by the FCT system initialization parameter.

**Attention:** If you use the **SHCDS REMOVESUBSYS** command for a CICS region that uses RLS access mode, ensure that you perform a cold start the next time you start the CICS region. The **SHCDS REMOVESUBSYS** command causes SMSVSAM to release all locks held for the region that is the subject of the command, allowing other CICS regions and batch jobs to update records released in this way. If you restart a CICS region with either a warm or emergency restart, after specifying it on a **REMOVESUBSYS** command, you risk losing data integrity.

You are recommended to use the REMOVESUBSYS command only for those CICS regions that you do not intend to run again, and therefore you need to free any retained locks that SMSVSAM might be holding.

**Terminal control resources:**

All previous terminal control information stored in the global catalog warm keypoint is lost.

Terminal control resource definitions are installed as follows:

**z/OS Communications Server devices**
: All Communications Server terminal resource definitions are installed from the CSD. The definitions to be installed are specified in groups named in the CSD group list, which is specified by the GRPLIST system initialization parameter. The resource definitions, of type TERMINAL and TYPETERM, include autoinstall model definitions as well as explicitly defined devices.

**Connection, sessions, and profiles**
: All connection and sessions definitions are installed from the CSD. The definitions to be installed are specified in groups named in the CSD group list, which is specified by the GRPLIST system initialization parameter. The connections and sessions resource definitions include those used for APPC autoinstall of parallel and single sessions, as well as explicitly defined connections.

**Sequential devices**
: Sequential (BSAM) device terminal resource definitions are installed from the terminal control table specified by the TCT system initialization parameter. CICS loads the table from the load library defined in the DFHRPL library concatenation. CICS TS for z/OS, Version 5.4

  Resource definitions for BSAM terminals are not cataloged at install time. They are cataloged only in the terminal control warm keypoint during a normal shutdown.

*Committing and cataloging resources installed from the CSD:*

CICS has two ways of installing and committing terminal resource definitions. Some resource definitions can be installed in groups or individually and are committed at the individual resource level, whereas some z/OS Communications Server (SNA) terminal control resource definitions must be installed in groups and are committed in "installable sets".

**Single resource install**
: All except the resources that are installed in installable sets are committed individually. CICS writes each single resource definition to the global catalog as the resource is installed. If a definition fails, it is not written to the catalog (and therefore is not recovered at a restart).

**Installable set install**
: The following Communications Server terminal control resources are committed in installable sets:

  - Connections and their associated sessions
  - Pipeline terminals:all the terminal definitions sharing the same POOL name

  If one definition in an installable set fails, the set fails. However, each installable set is treated independently within its CSD group. If an installable set fails as CICS installs the CSD group, it is removed from the set of successful installs. Logical sets that are not successfully installed do not have catalog records written and are not recovered.

If the installation of a resource or of an installable set is successful, CICS writes the resource definitions to the global catalog during commit processing.

### Distributed transaction resources

Unlike all other resources in a cold start, CICS preserves any information (units of work) about *distributed* transactions.

This action has no effect on units of work that relate only to the local CICS; it applies only to distributed units of work. The CICS recovery manager deals with these preserved units of work when resynchronization with the partner system takes place, just as in a warm or emergency restart.

This action is effective only if both the system log stream and the global catalog from the previous run of CICS are available at restart.

For information about recovery of distributed units of work, see Troubleshooting intersystem problems.

### Monitoring and statistics

The initial status of CICS monitoring is determined by the monitoring system initialization parameters (MN and MN*xxxx*).

The initial recording status for CICS statistics is determined by the **STATRCD** system initialization parameter. If STATRCD=ON is specified, interval statistics are recorded at the default interval of 1 hour.

## Starting CICS with the START=INITIAL parameter

If you specify START=INITIAL, CICS performs an initial start as if you are starting a new region for the first time.

### About this task

This initial start of a CICS region is different from a CICS region that initializes with a START=COLD parameter, as follows:

- The state of the global catalog is ignored. It can contain either data from a previous run of CICS, or it can be newly initialized. Any previous data is purged.
- The state of the system log is ignored. It can contain either data from a previous run of CICS, or it can reference new log streams. CICS does not keep any information saved in the system log from a previous run. The primary and secondary system log streams are purged and CICS begins writing a new system log.
- Because CICS is starting a new catalog, it uses a new logname token in the "exchange lognames" process when connecting to partner systems. Thus, remote systems are notified that CICS has performed a cold start and cannot resynchronize.
- User journals are not affected by starting CICS with the START=INITIAL parameter.

**Note:** An initial start can also result from a START=COLD parameter if the global catalog is newly initialized and does not contain a recovery manager control record. If the recovery manager finds that there is no control record on the catalog, it issues a message to the console prompting the operator to reply with a GO or CANCEL response. If the response is GO, CICS performs an initial start as if START=INITIAL was specified.

For more information about the effect of the state of the global catalog and the system log on the type of start CICS performs, see The role of the CICS catalogs.

## CICS warm restart

This section describes the CICS startup processing specific to a warm restart.

If you specify START=AUTO, which is the recommended method, CICS determines which type of start to perform using information retrieved from the recovery manager's control record in the global catalog. If the type-of-restart indicator in the control record indicates “warm start possible”, CICS performs a warm restart.

You should not attempt to compress a library after a warm start, without subsequently performing a CEMT SET PROGRAM(PRGMID) NEWCOPY for each program in the library. This is because on a warm start, CICS obtains the directory information for all programs which were installed on the previous execution. Compressing a library could alter its contents and subsequently invalidate the directory information known to CICS.

See CICS emergency restart for the restart processing performed if the type-of-restart indicates “emergency restart needed”.

### Rebuilding the CICS state after a normal shutdown

During a warm restart, CICS initializes using information from the catalogs and system log to restore the region to its state at the previous normal shutdown.

CICS needs both the catalogs and the system log from the previous run of CICS to perform a warm restart—the catalogs alone are not sufficient. If you run CICS with the system log defined as TYPE(DUMMY), CICS appears to shut down normally, but only the global catalog portion of the warm keypoint is written. Therefore, without the warm keypoint information from the system log, CICS cannot perform a warm restart. CICS startup fails unless you specify an initial start with START=INITIAL.

It is the responsibility of the individual resource managers (such as file control) and the CICS domains to recover their own state. The rebuilding process for resources varies depending on the type of resource, and whether or not the resource was defined as part of a CICS bundle.

#### Files

File control information from the previous run is recovered from information recorded in the CICS catalog only.

File resource definitions for VSAM and BDAM files, data tables, and LSR pools are installed from the global catalog, including any definitions that were added dynamically during the previous run. The information recovered and reinstalled in this way reflects the state of all file resources at the previous shutdown. For example:

- If you manually set a file closed (which changes its status to UNENABLED) and perform a normal shutdown, it remains UNENABLED after the warm restart.
- Similarly, if you set a file DISABLED, it remains DISABLED after the warm restart.

**Note:** An exception to this occurs when there are updates to a file to be backed out during restarts, in which case the file is opened regardless of the OPENTIME option. At a warm start, there cannot be any in-flight units of work to back out, so this backout can only occur when retrying backout-failed units of work against the file.

CICS closes all files at shutdown, and, as a general rule, you should expect your files to be re-installed on restart as either:

- OPEN and ENABLED if the OPENTIME option is STARTUP
- CLOSED and ENABLED if the OPENTIME option is FIRSTREF.

The FCT and the CSD*xxxx* system initialization parameters are ignored.

File control uses the system log to reconstruct the internal structures, which it uses for recovery.

**Data set name blocks:**

Data set name blocks (DSNBs), one for each data set opened by CICS file control, are recovered during a warm restart.

If you have an application that creates many temporary data sets, with a different name for every data set created, it is important that your application removes these after use. If applications fail to get rid of unwanted name blocks they can, over time, use up a considerable amount of CICS dynamic storage. You can use the **SET DSNAME REMOVE** command to remove unwanted data set name blocks.

**Reconnecting to SMSVSAM for RLS access:**

CICS connects to the SMSVSAM server, if present, and exchanges RLS recovery information.

In this exchange, CICS finds out whether SMSVSAM has lost any retained locks while CICS has been shut down. This could happen, for example, if SMSVSAM could not recover from a coupling facility failure that caused the loss of the lock structure. If this has happened, CICS is notified by SMSVSAM to perform **lost locks recovery**. See Lost locks recovery for information about this process.

**Recreating non-RLS retained locks:**

For non-RLS files, the CICS enqueue domain rebuilds the retained locks relating to shunted units of work.

## Temporary storage

Auxiliary temporary storage queue information (for both recoverable and non-recoverable queues) is retrieved from the warm keypoint. Note that TS READ pointers are recovered on a warm restart (which is not the case on an emergency restart).

CICS opens the auxiliary temporary storage data set for update.

**Temporary storage data sharing server:**

Any queues written to a shared temporary storage pool, even though non-recoverable, persist across a warm restart.

## Transient data

Transient data initialization on a warm restart depends on the TDINTRA system initialization parameter, which specifies whether or not TD is to initialize with empty intrapartition queues. The different options are discussed as follows:

**TDINTRA=NOEMPTY (the default):**

All transient data resource definitions are installed from the global catalog, including any definitions that were added dynamically during the previous run. TD queues are always installed as enabled.

CICS opens any extrapartition TD queues that need to be opened—that is, any that specify OPEN=INITIAL.

**Note:** If, during the period when CICS is installing the TD queues, an attempt is made to write a record to a CICS-defined queue that has not yet been installed (for example, CSSL), CICS writes the record to the CICS-defined queue CXRF.

The recovery manager returns log records and keypoint data associated with TD queues. CICS applies this data to the installed queue definitions to return the TD queues to the state they were in at normal shutdown. Logically recoverable, physically recoverable, and non-recoverable intrapartition TD queues are recovered from the warm keypoint data.

*Trigger levels (for TERMINAL and SYSTEM only):*

After the queues have been recovered, CICS checks the trigger level status of each intrapartition TD queue that is defined with FACILITY(TERMINAL|SYSTEM) to determine whether a start request needs to be rescheduled for the trigger transaction.

If a trigger transaction failed to complete during the previous run (that is, did not reach the empty queue (QZERO) condition) or the number of items on the queue is greater than the trigger level, CICS schedules a start request for the trigger transaction.

This does not apply to trigger transactions defined for queues that are associated with files (FACILITY(FILE)).

**TDINTRA=EMPTY:**

If you specify this option, a cold start will be performed for the transient data queues, but the resource definitions are warm started.

The following processing takes place:

- All intrapartition TD queues are initialized empty.
- The queue resource definitions are installed from the global catalog, but they are not updated by any log records or keypoint data. They are always installed enabled.

This option is intended for use when initiating remote site recovery (see CICS emergency restart), but you can also use it for a normal warm restart. For example, you might want to 'cold start' the intrapartition queues when switching to a new data set if the old one is corrupted, while preserving all the resource definitions from the catalog.

You cannot specify a general cold start of transient data while the rest of CICS performs a warm restart, as you can for temporary storage.

## Transactions

All transaction and transaction class resource definitions are installed from the CSD, and updated with information from the warm keypoint in the system log. The resource definitions installed from the catalog include any that were added dynamically during the previous run.

## LIBRARY resources

On WARM or EMERGENCY start, all LIBRARY definitions will be restored from the catalog, and the actual search order through the list of LIBRARY resources that was active at the time of the preceding shutdown will be preserved.

The latter will ensure that the search order of two LIBRARY resources of equal RANKING will remain the same. An equal RANKING implies that the relative search order of the LIBRARY resources is unimportant, but unexpected behavior might result if this order changed after a warm or emergency restart.

If a LIBRARY with an option of CRITICAL(YES) is restored from the catalog, and one of the data sets in its concatenation is no longer available, a message will be issued to allow the operator to choose whether to continue the CICS startup, or to cancel it. This Go or Cancel message will be preceded by a set of messages providing information on any data sets which are not available. For LIBRARY resources, with an option of CRITICAL(NO), this condition will not cause CICS startup to fail, but a warning message will be issued and the LIBRARY will not be reinstalled. This warning message will be preceded by a set of messages providing information on any data sets which are not available

## Programs

The recovery of program, mapset, and partitionset resource definitions depends on whether you are using program autoinstall and, if you are, whether you have requested autoinstall cataloging (specified by the system initialization parameter PGAICTLG=ALL|MODIFY).

**No autoinstall for programs:**

If program autoinstall is disabled (PGAIPGM=INACTIVE), all program, mapset, and partitionset resource definitions are installed from the CSD, and updated with information from the warm keypoint in the system log.

The resource definitions installed from the catalog include any that were added dynamically during the previous run.

**Autoinstall for programs:**

If program autoinstall is enabled (PGAIPGM=ACTIVE), program, mapset, and partitionset resource definitions are installed from the CSD only if they were cataloged; otherwise they are installed at first reference by the autoinstall process.

All definitions installed from the CSD are updated with information from the warm keypoint in the system log.

CICS catalogs program, mapset, and partitionset resource definitions as follows:

- If they are installed from predefined definitions in the CSD, either during a cold start or by an explicit INSTALL command, CICS catalogs the definitions.

- If the PGAICTLG system initialization parameter specifies ALL, CICS catalogs all the autoinstalled program-type definitions, and these are reinstalled during the warm restart.
- If the PGAICTLG system initialization parameter specifies MODIFY, CICS catalogs only those autoinstalled program-type definitions that are modified by a SET PROGRAM command, and these are reinstalled during the warm restart.
- CICS never catalogs programs that are autoinstalled by a task for an application that is deployed on a platform, regardless of the setting for the PGAICTLG system initialization parameter, so these programs are not reinstalled during the warm restart.

## Start requests

In general, start requests are recovered together with any associated start data.

Recovery can, however, be suppressed by specifying explicit cold start system initialization parameters for temporary storage, interval control, or basic mapping support (on the TS, ICP, and BMS system initialization parameters respectively). Any data associated with suppressed starts is discarded.

The rules governing the operation of the explicit cold requests on system initialization parameters are:

- ICP=COLD suppresses all starts that do not have both data and a terminal associated with them. It also suppresses any starts that had not expired at shutdown. This includes BMS starts.
- TS=COLD (or TS main only) suppresses all starts that had data associated with them.
- BMS=COLD suppresses all starts relating to BMS paging.

Start requests that have not been suppressed for any of the preceding reasons either continue to wait if their start time or interval has not yet expired, or they are processed immediately. For start requests with terminals, consider the effects of the CICS restart on the set of installed terminal definitions. For example, if the terminal specified on a start request is no longer installed after the CICS restart, CICS invokes an XALTENF global user exit program (if enabled), but not the XICTENF exit.

## Monitoring and statistics

The CICS monitoring and statistics domains retrieve their status from their control records stored in the global catalog at the previous shutdown.

This is modified by any runtime system initialization parameters.

## Journal names and journal models

The CICS log manager restores the journal name and journal model definitions from the global catalog. Journal name entries contain the names of the log streams used in the previous run, and the log manager reconnects to these during the warm restart.

## Terminal control resources

Terminal control information is installed from the warm keypoint in the global catalog, or installed from the terminal control table (TCT), depending on whether the resources are CSD-defined or TCT-defined.

**CSD-defined resource definitions:**

When resources are defined in the CICS System Definition data set (CSD), terminal control information is installed from the warm keypoint in the global catalog.

CICS installs the following terminal control resource definitions from the global catalog:

- All permanent terminal devices, originally installed from explicit resource definitions, and profiles.
- The following autoinstalled APPC connections:
  - Synclevel-2-capable connections (for example, CICS-to-CICS connections)
  - Synclevel-1-capable, limited resource connections installed on a CICS that is a member of a z/OS Communications Server generic resource.

  Other autoinstalled terminals are not recovered, because they are removed from the warm keypoint during normal shutdown. This ensures that their definitions are installed only when terminal users next log on after a CICS restart that follows a normal shutdown.

When a multiregion operation (MRO) connection is restored, it has the same status that was defined in the CSD. Any changes of status, for example the service status, are not saved on the global catalog, so are not recovered during a warm or emergency restart.

Only the global catalog is referenced for terminals defined in the CSD.

To add a terminal after initialization, use the CEDA INSTALL or EXEC CICS CREATE command, or the autoinstall facility. To delete a terminal definition, use the DISCARD command or, if autoinstalled, allow it to be deleted by the autoinstall facility after the interval specified by the AILDELAY system initialization parameter.

**Sequential (BSAM) devices:**

Terminal control information for sequential terminal devices is installed from the terminal control table (TCT).

CICS installs sequential terminal resource definitions as follows:

- **Same TCT as last run**. CICS installs the TCT and then modifies the terminal entries in the table by applying the cataloged data from the terminal control warm keypoint from the previous shutdown. This means that, if you reassemble the TCT and keep the same suffix, any changes you make could be undone by the warm keypoint taken from the catalog.
- **Different TCT from last run**. CICS installs the TCT only, and does not apply the warm keypoint information, effectively making this a cold start for these devices.

## Distributed transaction resources

CICS retrieves its logname from the recovery manager control record in the global catalog for use in the “exchange lognames” process with remote systems. Resynchronization of indoubt units of work takes place after CICS completes reconnection to remote systems.

See Recovery functions and interfaces for information about recovery of distributed units of work.

## URIMAP definitions and virtual hosts

Installed URIMAP definitions for CICS Web support are restored from the global catalog, including their enable status. Virtual hosts, which are created by CICS using the host names specified in installed URIMAP definitions, are also restored to their former enabled or disabled state.

## Recovery of resources in bundles

Resources that are defined in CICS bundles are not stored in the CSD. BUNDLE resources can be created from a definition and these resources are stored in the catalog. BUNDLE resources can also be created when you install platforms and applications. These resources have no definition, are not stored in the catalog, and are recovered through the application or platform.

### Recovery for BUNDLE definitions

On a cold start of CICS, BUNDLE resource definitions are deleted from the catalog and the bundle is not re-created.

On a warm or emergency restart of CICS, during post-initialization processing CICS tries to re-create all the BUNDLE resource definitions that were installed before the restart, and install them in the enablement state that they were in when the CICS region stopped.

For BUNDLE resources that were created from a definition and stored in the catalog, CICS uses the CRLR supplied transaction to start a program that resolves all of the resources that are defined in the bundle manifest, including the dynamic re-creation of all the required CICS resources. Although most dynamically created resources are not defined in the catalog, EVENTBINDING, EPADAPTER, and EPADAPTERSET resources are stored in the catalog.

If CICS is unable to create and enable a resource dynamically, the BUNDLE resource installs in the disabled state and a warning message is issued. Even if only one out of a number of resources fail to install, the BUNDLE resource installs in a disabled state. Use the Bundle Parts view in the CICS Explorer to view the state of every resource in a BUNDLE resource.

For a standalone BUNDLE resource that contains application entry points, the availability status of the bundle is also recovered during the restart of a CICS region.

### Recovery for BUNDLE resources generated by applications and platforms

When you install a platform in a CICSplex, or install an application in a platform, any CICS bundles that are part of the deployment are dynamically created in the appropriate CICS regions by CICSPlex SM. Each BUNDLE resource is dynamically created and is given a unique name. Each BUNDLE resource also has a BASESCOPE value that contains the name of the platform, the application, and the application version. Because the BUNDLE resources are dynamically created, they do not have equivalent definitions in the CSD and are not stored in the catalog.

The resources that are defined inside each CICS bundle for a platform or application are dynamically created in the CICS regions during the installation of the dynamically created BUNDLE resource. These resources also do not have equivalent definitions in the CSD, and are not stored in the catalog.

When you install an application or platform, CICSPlex SM creates a record for the platform or application in the data repository, which is used in recovery processing for the CICS bundles associated with the platform or application. The CICS bundles for applications are initially installed in the CICS regions in a disabled state. When you enable the application, CICSPlex SM enables the bundles in the CICS regions. When you make the application available, callers can invoke the application through its application entry points.

When you start or restart a CICS region that is defined as part of a platform, CICSPlex SM reads the information in the platform and application bundles in zFS, and installs the appropriate versions of the appropriate CICS bundles into the CICS region. The CICS bundles are installed in the same way during a cold, warm, or emergency restart of a CICS region. The generated resources inside the bundles are installed regardless of the autoinstall status or cataloguing for similar resources in the CICS region that are not part of an application bundle.

- CICS bundles associated with a platform are installed in the same state as the platform, unless there is a problem with the installation. If the platform is disabled, the CICS bundles associated with it are disabled, and if the platform is enabled, the bundles are enabled
- If an application version was not enabled before the start or restart of the CICS region, its CICS bundles are installed in a disabled state. If an application version was enabled, its CICS bundles are installed in an enabled state, unless there is a problem with the installation.
- The availability status of an application version is applied at the start of a CICS region, and is recovered during a restart of a CICS region.

The relationship between a CICS bundle and the CICS regions where it is installed is stored in a management part for the application or platform. The management part is a MGMTPART record that is created automatically during the application install process. The overall status information for an application is derived from the status of the individual management parts for the application. The overall status information for a platform is derived from the status of the region types and the status of the individual management parts for the platform. To check the status for an application or platform, use the Cloud Explorer view in the CICS Explorer.

### Recovery of user resources

If a bundle contains a resource type that is handled outside CICS, for example a vendor resource, the bundle registration program must be available during post-initialization programming to register the callback program and re-create the resource type.

If the registration program does not register the callback program or the callback program is not available to re-create the user resources, the BUNDLE resource installs in the disabled state and the user resources install in the unusable state. You must ensure that both the registration and callback programs are available in CICS before discarding and re-creating the BUNDLE resources.

### Recovery of files in bundles

When a file that is defined in a CICS bundle is installed, it is added to the catalog. CICS recovers the bundle installed files from the catalog during a warm or emergency restart. These recovered files are used for the CICS recovery only. After CICS completes the recovery, all these files are deleted. When the bundle is re-created, CICS picks up the definition from the bundle directory, and creates a

new file. If a file recovered from the catalog has a retained lock, it cannot be deleted on completion of recovery, and as a result, the file becomes orphaned (does not belong to any bundle) and has to be deleted manually after CICS has completed restart.

## Automatic restart management

CICS uses the automatic restart manager (ARM) component of MVS to increase the availability of your systems.

MVS automatic restart management is a sysplex-wide integrated automatic restart mechanism that performs the following tasks:

- Restarts an MVS subsystem in place if it abends (or if a monitor program notifies ARM of a stall condition)
- Restarts all the elements of a workload (for example, CICS TORs, AORs, FORs, DB2) on another MVS image after an MVS failure
- Restarts CICS data sharing servers in the event of a server failure.
- Restarts a failed MVS image

CICS reconnects to DBCTL and the z/OS Communications Server automatically if either of these subsystems restart after a failure. CICS is not dependent on using ARM to reconnect in the event of failure.

The MVS automatic restart manager provides the following benefits:

- Enables CICS to preserve data integrity automatically in the event of any system failure.
- Eliminates the need for operator-initiated restarts, or restarts by other automatic packages, thereby:
  - Improving emergency restart times
  - Reducing errors
  - Reducing complexity.
- Provides cross-system restart capability. It ensures that the workload is restarted on MVS images with spare capacity, by working with the MVS workload manager.
- Allows all elements within a restart group to be restarted in parallel. Restart levels (using the ARM WAITPRED protocol) ensure the correct starting sequence of dependent or related subsystems.

### Restrictions

MVS automatic restart management is available only to those MVS subsystems that register with ARM. CICS regions register with ARM automatically as part of CICS system initialization. If a CICS region fails before it has registered for the first time with ARM, it will not be restarted. After a CICS region has registered, it is restarted by ARM according to a predefined policy for the workload.

## CICS ARM processing

A prime objective of CICS support for the MVS automatic restart manager (ARM) is to preserve data integrity automatically in the event of any system failure.

If CICS is restarted by ARM with the same persistent JCL, CICS forces START=AUTO to ensure data integrity.

## Registering with ARM

To register with ARM, you must implement automatic restart management on the MVS images that the CICS workload is to run on. You must also ensure that the CICS startup JCL used to restart a CICS region is suitable for ARM.

### Before you begin

The implementation of ARM is part of setting up your MVS environment to support CICS. See Implementing MVS automatic restart management in Installing.

### About this task

During initialization, CICS registers with ARM automatically.

CICS always registers with ARM because CICS needs to know whether it is being restarted by ARM and, if it is, whether the restart is with persistent JCL. (The ARM registration response to CICS indicates whether the same JCL that started the failed region is being used for the ARM restart.) You indicate whether MVS is to use the same JCL or command text that previously started CICS by specifying PERSIST as the *restart_type* operand on the **RESTART_METHOD** parameter in your automatic restart management policy.

When it registers with ARM, CICS passes the value SYSCICS as the element type, and the string SYSCICS_*aaaaaaaa* as the element name, where *aaaaaaaa* is the CICS applid. Because the applid is used in the element name, only one CICS region can successfully register with ARM for a given applid. If two CICS regions try to register with the same applid, ARM rejects the second region.

## Waiting for predecessor subsystems

During initialization CICS issues an ARM WAITPRED (**wait predecessor**) request to wait, if necessary, for predecessor subsystems (such as DB2 and DBCTL) to become available.

This is indicated by message DFHKE0406. One reason for this wait is to ensure that CICS can resynchronize with its partner resource managers for recovery purposes before accepting new work from the network.

## Unregistering from ARM

During normal shutdown, unless you specify the RESTART option on the **PERFORM SHUT** command, CICS unregisters from ARM to ensure that it is not automatically restarted. Also, if you want to perform an immediate shutdown and do not want ARM to cause an automatic restart, you can specify the NORESTART option on **PERFORM SHUT IMMEDIATE**.

### About this task

Some error situations that occur during CICS initialization cause CICS to issue a message, with an operator prompt to reply GO or CANCEL. If you reply CANCEL, CICS de-registers from ARM before terminating, because if CICS remained registered, an automatic restart would probably encounter the same error condition.

For other error situations, CICS does not unregister, and automatic restarts follow. To control the number of restarts, specify in your ARM policy the number of times ARM is to restart a failed CICS region.

### Failing to register

If ARM support is present but the register fails, CICS issues message DFHKE0401. In this case, CICS does not know if it is being restarted by ARM, and therefore it doesn't know whether to override the START parameter to force an emergency restart to preserve data integrity.

If START=COLD or START=INITIAL is specified as a system initialization parameter and CICS fails to register, CICS also issues message DFHKE0408. When CICS is restarting with START=COLD or START=INITIAL, CICS relies on ARM to determine whether to override the start type and change it to AUTO. Because the REGISTER has failed, CICS cannot determine whether the region is being restarted by ARM, and so does not know whether to override the start type. Message DFHKE0408 prompts the operator to reply ASIS or AUTO, to indicate the type of start CICS is to perform:

- A reply of ASIS means that CICS is to perform the start specified on the START parameter.
- A reply of AUTO means that CICS is being restarted by ARM, and the type of start is to be resolved by CICS. If the previous run terminated abnormally, CICS will perform an emergency restart.

  **Note:** A CICS restart can have been initiated by ARM, even though CICS registration with ARM has failed in the restarted CICS.

### ARM couple data sets

You must ensure that you define the couple data sets required for ARM and that they are online and active before you start any CICS region for which you want ARM support.

- CICS automatic ARM registration fails if the couple data sets are not active at CICS startup. When CICS is notified by ARM that registration has failed for this reason, CICS assumes this means that you do not want ARM support, and CICS initialization continues.
- If ARM loses access to the couple data sets, the CICS registration is lost. In this event, ARM cannot restart a CICS region that fails.

See *z/OS MVS Setting Up a Sysplex* for information about ARM couple data sets and ARM policies.

## CICS restart JCL and parameters

Each CICS restart can use the previous startup JCL and system initialization parameters, or can use a new job and parameters.

### CICS START options

You are recommended to specify START=AUTO, which causes a warm start after a normal shutdown and an emergency restart after failure.

You are also recommended always to use the same JCL, even if it specifies START=COLD or START=INITIAL, to ensure that CICS restarts correctly when restarted by the MVS automatic restart manager after a failure.

If you specify START=COLD (or INITIAL) and your ARM policy specifies that the automatic restart manager is to use the same JCL for a restart following a CICS failure, CICS overrides the start parameter when restarted by ARM and enforces

START=AUTO. CICS issues message DFHPA1934 and ensures the resultant emergency restart handles recoverable data correctly.

If the ARM policy specifies different JCL for an automatic restart and that JCL specifies START=COLD, CICS uses this parameter value but risks losing data integrity. Therefore, if you need to specify different JCL to ARM, specify START=AUTO to ensure data integrity.

## Workload policies

Workloads are started initially by scheduling or automation products. The components of the workload, and the MVS images capable of running them, are specified as part of the policies for z/OS Workload Manager and ARM.

The MVS images must have access to the databases, logs, and program libraries required for the workload.

Administrative policies provide ARM with the necessary information to perform appropriate restart processing. You can define one or more administrative policies, but can have only one active policy for all MVS images in a sysplex. You can modify administrative policies by using an MVS-supplied utility, and can activate a policy with the MVS SETXCF command.

## Connecting to the z/OS Communications Server

The z/OS Communications Server for SNA is at restart level 1, the same as DB2 and DBCTL.

However, the Communications Server is not restarted when failed subsystems are being restarted on another MVS, because ARM expects the Communications Server to be running on all MVS images in the sysplex. For this reason, CICS and the Communications Server are not generally part of the same restart group.

In an SNA network, the session between CICS and the Communications Server is started automatically if the Communications Server is started before CICS. If the Communications Server is not active when you start (or restart) CICS, you receive the following messages:

```
+DFHSI1589D 'applid' VTAM is not currently active.
+DFHSI1572 'applid'  Unable to OPEN VTAM ACB - RC=xxxxxxxx, ACB CODE=yy.
```

**Note:** VTAM is now the z/OS Communications Server.

CICS provides transaction COVR to open the SNA ACB automatically when the Communications Server becomes available.

### The COVR transaction

To ensure that CICS reconnects to the z/OS Communications Server in the event of a Communications Server abend, CICS keeps retrying the OPEN VTAM ACB using a time-delay mechanism via the non-terminal transaction COVR.

After CICS has completed clean-up following the Communications Server failure, it invokes the CICS open z/OS Communications Server retry (COVR) transaction. The COVR transaction invokes the terminal control open z/OS Communications Server retry program, DFHZCOVR, which performs an OPEN VTAM retry loop with a 5-second wait. CICS issues a DFHZC0200 message every minute, while the open is unsuccessful, and each attempt is recorded on the CSNE transient data queue. After ten minutes, CICS issues a DFHZC0201 message and terminates the

transaction. If CICS shutdown is initiated while the transaction is running, CICS issues a DFHZC0201 message and terminates the transaction.

You cannot run the COVR transaction from a terminal. If you invoke COVR from a terminal, it abends with an AZCU transaction abend.

**Note:** VTAM is now the z/OS Communications Server.

### Messages associated with automatic restart

There are some CICS messages for ARM support, which CICS can issue during startup if problems are encountered when CICS tries to connect to ARM.

The message numbers are:

DFHKE0401
DFHKE0402
DFHKE0403
DFHKE0404
DFHKE0405
DFHKE0406
DFHKE0407
DFHKE0408
DFHKE0410
DFHKE0411
DFHZC0200
DFHZC0201

## Automatic restart of CICS data-sharing servers

All three types of CICS data-sharing server—temporary storage, coupling facility data tables, and named counters—support automatic restart using the services of automatic restart manager.

The servers also have the ability to wait during start-up, using an event notification facility (ENF) exit, for the coupling facility structure to become available if the initial connection attempt fails.

### Server ARM processing

During initialization, a data-sharing server unconditionally registers with ARM, except when starting up for unload or reload. A server does not start if registration fails, with return code 8 or higher.

If a server encounters an unrecoverable problem with the coupling facility connection, consisting either of lost connectivity or a structure failure, it cancels itself using the server command `CANCEL RESTART=YES`. This terminates the existing connection, closes the server and its old job, and starts a new instance of the server job.

You can also restart a server explicitly using either the server command `CANCEL RESTART=YES`, or the MVS command `CANCEL` *`jobname`*`,ARMRESTART`

By default, the server uses an ARM element type of SYSCICSS, and an ARM element identifier of the form DFH*xxnn_poolname* where *xx* is the server type (XQ, CF or NC) and *nn* is the one- or two-character &SYSCLONE identifier of the MVS

image. You can use these parameters to identify the servers for the purpose of overriding automatic restart options in the ARM policy.

**Waiting on events during initialization:**

If a server is unable to connect to its coupling facility structure during server initialization because of an environmental error, the server uses an ENF event exit to wait for cross-system extended services (XES) to indicate that it is worth trying again.

The event exit listens for either:

- A specific XES event indicating that the structure has become available, or
- A general XES event indicating that some change has occurred in the status of coupling facility resources (for example, when a new CFRM policy has been activated).

When a relevant event occurs, the server retries the original connection request, and continues to wait and retry until the connection succeeds. A server can be canceled at this stage using an MVS `CANCEL` command if necessary.

**Server initialization parameters for ARM support:**

The server startup parameters for ARM support are:

**ARMELEMENTNAME=***elementname*
: specifies the automatic restart manager element name, up to 16 characters, to identify the server to ARM for automatic restart purposes.

**ARMELEMENTTYPE=***elementtype*
: specifies the automatic restart manager element type, up to 8 characters for use in ARM policies as a means of classifying similar elements.

These parameters are the same for all the data sharing servers. For more details, see Automatic restart manager (ARM) parameters.

**Server commands for ARM support:**

The following are the ARM options you can use on server commands:

**CANCEL RESTART={NO|YES}**
: terminates the server immediately, specifying whether or not automatic restart should be requested. The default is RESTART=NO.

  You can also enter RESTART on its own for RESTART=YES, NORESTART for RESTART=NO.

**ARMREGISTERED**
: shows whether ARM registration was successful (YES or NO).

**ARM**
: This keyword, in the category of display keywords that represent combined options, can be used to display all ARM-related parameter values. It can also be coded as **ARMSTATUS**.

These commands are the same for all the data sharing servers.

## Backup-while-open (BWO)

The BWO facility, together with other system facilities and products, allows you to take a backup copy of a VSAM data set while it remains open for update.

Many CICS applications depend on their data sets being open for update over a long period of time. Normally, you cannot take a backup of the data set while the data set is open. Thus, if a failure occurs that requires forward recovery, all updates that have been made to the data set since it was opened must be recovered. This means that you must keep all forward recovery logs that have been produced since the data set was opened. A heavily used data set that has been open for update for several days or weeks might need much forward recovery.

Using BWO, only the updates that have been made since the last backup copy was taken need to be recovered. This could considerably reduce the amount of forward recovery that is needed.

### BWO and concurrent copy

Concurrent copy improves BWO processing by eliminating the invalidation of a BWO dump because of updates to the data set.

The following is a comparison of various kinds of dumps that you can request:

- **Normal dump.** Use of the data set must be quiesced so that serialization is obtained, the data set is dumped, and serialization is released. The data set cannot be used for the entire time.
- **Concurrent copy dump**. Use of the data set must be quiesced so that serialization is obtained, concurrent copy utilization is completed within a very short time (compared with the actual time to dump the data set), serialization is released, and the data set is dumped. The data set can be used after concurrent copy initialization is complete.
- **BWO dump**. Serialization is attempted but is not required, and the data set is dumped. If it is eligible for BWO, the data set is dumped without serialization and can remain in use for the entire time, but the dump can be invalidated by update activity to the data set.
- **BWO dump using concurrent copy**. Serialization is attempted but is not required, concurrent copy initialization is completed, and the data set is dumped. If it is eligible for BWO, the data set is dumped without serialization and can remain in use for the entire time, and updates that occur do not cause the dump to be invalidated.

To use concurrent copy, specify the CONCURRENT keyword when you use DFSMShsm to dump BWO data sets.

#### BWO and backups

The BWO function allows backups to be taken by DFSMSdss when applications are running in continuous operation while the data set is open for update with full data integrity of copied data.

Continuous operation is typically 24 hours-a-day for five to seven days a week. This is feasible only for CICS VSAM file control data sets for which CICS creates forward-recovery logs. Long-running transactions, automated teller machines, and continuously available applications require the database to be up and running when the backup is being taken.

The concurrent copy function used along with BWO by DFSMSdss allows backups to be taken with integrity even when control-area and control-interval splits and data set additions (new extents or add-to-end) are occurring for VSAM key sequenced data sets.

## BWO requirements

To use the backup-while-open (BWO) support provided by CICS, you can use the Data Facility Storage Management Subsystem/MVS (DFSMS/MVS) or a licensed program that provides equivalent function.

You must have your environment configured with the following modules and components:

- Use Release 2, or later, for data sets used in non-RLS access mode and Release 3 for data sets used in RLS access mode.
- You must install the DFSMSdfp IGWAMCS2 callable services module in the link pack area (LPA).
- You must install the IGWABWO module, supplied in SYS1.CSSLIB, in the LPA, or include SYS1.CSSLIB in the link list. Do not include the library in the STEPLIB or JOBLIB library concatenations.
- You must have the DFSMSdfp, DFSMSdss, and DFSMShsm components of DFSMS installed on the processors that perform backup and recovery.

During initialization, CICS determines the availability of BWO support by issuing calls to the callable services modules IGWAMCS2 and IGWABWO. CICS also checks on the DFSMSdss release level by calling the DFSMSdss module ADRRELVL. If access to this DFSMSdss module is strictly controlled by an external security manager, such as RACF®, security violation messages are issued against the CICS userid, unless the CICS region userid is authorized to access this module.

Note that CICS VSAM Recovery for z/OS, which performs forward recovery, must be installed on the processor where forward recovery is to be done. CICS VSAM Recovery is required for forward recovery and backout of CICS VSAM data sets backed up with BWO and concurrent copy functions of DFSMS/MVS.

Table 3 cross-references the storage management component names of DFSMS to the previous names of the individual licensed programs:

*Table 3. Cross-reference of DFSMS/MVS product terminology*

| Component name | Full DFSMS/MVS name | Previous product name |
|---|---|---|
| DFSMSdfp | Data Facility Storage Management Subsystem data facility product | MVS/DFP |
| DFSMShsm | Data Facility Storage Management Subsystem hierarchical storage manager | DFHSM |
| DFSMSdss | Data Facility Storage Management Subsystem data set services | DFDSS |

### Hardware requirements

The concurrent copy function is supported by the IBM 3990 Model 3 with the extended platform and the IBM 3990 Model 6 control units.

## Which data sets are eligible for BWO

You can use BWO only for:

- Data sets that are on SMS-managed storage and that have an integrated catalog facility (ICF) catalog.
- VSAM data sets accessed by CICS file control and for the CICS system definition (CSD) file. ESDS, KSDS, and RRDS are supported. ESDS and KSDS are supported both with and without alternate indexes. DFSMShsm imposes a limit (many thousands) on the number of alternate indexes for a data set.

BWO is supported at the VSAM sphere level; thus you cannot take BWO backup copies of some sphere components and not others. The first data set opened for update against a VSAM base cluster determines the BWO eligibility for the sphere. This includes base clusters that are accessed through a VSAM path key. For example, if the first data set is defined as eligible for BWO, CICS fails the file-open operation for any subsequent data set that is opened for update against that cluster and which is not defined as eligible for BWO.

You can take BWO volume backups if all data sets that are open for update on the volume are eligible for BWO.

### VSAM control interval or control area split

For a KSDS (or an ESDS with alternate indexes) that has frequent insert or update activity, it is only practical (and safe) to take BWO backups during periods of reduced activity (for example, overnight). This is because it is possible for a VSAM control interval or control area split to occur during a BWO backup. Then, the integrity of the backup copy cannot be guaranteed because DFSMSdss copies the data set sequentially, and so certain portions of the data set might be duplicated or not represented at all in the backup copy.

DFSMSdfp indicates in the ICF catalog that a split has occurred. DFSMShsm and DFSMSdss check the ICF catalog at the start and end of a backup. If a split is in progress at the start of a backup, the backup is not taken. If a split has occurred during a backup, or a split is still in progress at the end of a backup, the backup is discarded.

So, to take a BWO backup successfully, the normal time between splits must be greater than the time taken for DFSMShsm and DFSMSdss to take a backup of the data set.

***Data tables***: You can use BWO with CICS-maintained data table base clusters. However, you cannot use BWO with user-maintained data tables, because no forward recovery support is provided.

***Alternate Index***: CICS normally uses a base key or a path key to access data in a VSAM base cluster data set. It is also possible, but not normal, for CICS to access alternate index records by specifying the alternate index name as the data set name. If an alternate index data set is used in this way, you cannot define the alternate index as eligible for BWO. Instead, the alternate index adopts the BWO characteristics already defined for the VSAM sphere.

## How you request BWO

You can define files as eligible for BWO in one of two ways.

## About this task

The method that you can use to define files as eligible for BWO depends on the mode in which the files are accessed.

Do not define BWO for the CICSPlex SM data repository using the IDCAMS DEFINE CLUSTER definition within the ICF catalog because performance is degraded. See Defining a forward recovery log for the data repository for information on taking backups of the CICSPlex SM data repository.

## Procedure

Decide which method you want to use for data sets:

- If your data set is accessed in RLS mode, you must define the BWO option in the ICF catalog. Defining BWO in the ICF catalog requires DFSMS 1.3.
- If your data set is accessed only in non-RLS mode, you can define the BWO option in either the ICF catalog or the CICS file definition.

Defining BWO in the ICF catalog requires DFSMS 1.3. For data sets that are accessed in RLS mode, the BWO option must be defined in the ICF catalog. Recovery attributes for data sets that are accessed only in non-RLS mode can be defined either in the ICF catalog or in the CICS FILE resource. If BWO is defined in the ICF catalog definition, by default it overrides any BWO option defined in the FILE resource. To force CICS to use the FILE resource attributes instead of the catalog recovery options, set the **NONRLSRECOV** system initialization parameter to FILEDEF.

## Specifying BWO using access method services

There is a BWO parameter on the access method services DEFINE CLUSTER statement.

### About this task

You can specify the BWO parameter as follows:

**TYPECICS**
The data set is eligible for BWO in CICS.

**NO** The data set is not eligible for BWO.

**TYPEIMS**
The data set is eligible for BWO in IMS™, but CICS treats this as NO.

**TYPEOTHER**
The data set is eligible for BWO, but CICS treats this as NO.

If you specify BWO(TYPECICS), you might need to know whether the PTF associated with APAR OA04220 has been applied to your z/OS system. That APAR extended the capabilities of VSAM so that BWO processing for RLS is no longer disabled for non-recoverable files.

- If you specify BWO(TYPECICS), and you have also specified LOG(ALL) and a forward recovery log stream name, LOGSTREAMID(*logstream_name*), your file is recoverable, and the status of the PTF is not important.
- If you specify BWO(TYPECICS), and the PTF has been applied, BWO processing is now enabled for all VSAM RLS files whether you have specified LOG(ALL), LOG(NONE) or LOG(UNDO).

- But if you specify BWO(TYPECICS), and the PTF has not been applied, and you have not specified LOG(ALL) and a forward recovery log stream name, BWO processing for RLS remains disabled for such files. To achieve BWO for the file, you must either:
  - apply the PTF,
  - or specify LOG(ALL) and a forward recovery log stream name (if those actions are appropriate for the file in question).

If you omit the BWO parameter from the DEFINE statement, by default it is UNDEFINED in the ICF catalog, and the BWO attribute from the CICS file resource definition is used.

BWO(TYPECICS) is the equivalent of BACKUPTYPE(DYNAMIC) in a CICS file resource definition. All other values, including UNDEFINED, are treated by CICS as the equivalent of BACKUPTYPE(STATIC) in a CICS file resource definition. For simplicity, the CICS terms BACKUPTYPE(DYNAMIC) and BACKUPTYPE(STATIC) are used unless it is necessary to specifically mention the access method services BWO parameters.

The BWO options for the CSD are taken from the ICF catalog if they are defined there, and the system initialization parameters (CSDBKUP, CSDRECOV, and CSDFRLOG) are ignored.

## Specifying BWO on CICS file resource definitions

You define a file as eligible for BWO with the BACKUPTYPE attribute on a CICS FILE resource definition in the CSD.

### About this task

If you specify BACKUPTYPE(DYNAMIC), the file is defined as eligible for BWO when the data set is opened. You must also specify RECOVERY(ALL) and FWDRECOVLOG(*nn*) to request forward recovery support.

BACKUPTYPE(STATIC), the default, defines a file as not eligible for BWO. In this case, if DFSMShsm is to back up a data set, all CICS files currently open for update against that data set must be closed before the backup can start.

All files opened against the same VSAM base cluster must have the same BACKUPTYPE value. That value is established by the first file opened against the cluster; it is stored in the CICS data set name block (DSNB) for the cluster. If the value for a subsequent file does not match, the file-open operation fails.

The BACKUPTYPE value in the DSNB persists across warm and emergency restarts. It is removed by a CICS cold start (unless a backout failure occurs) or by issuing EXEC CICS SET DSNAME ACTION(REMOVE) (or the CEMT equivalent) for the base cluster data set. To do this, all files that are open against the base cluster and via path definition must be closed, and the DSNB must have FILECOUNT of zero and NORMALBKOUT state.

The BACKUPTYPE attribute is ignored for user-maintained data table base clusters, because no forward recovery support is provided.

To use BWO for the CSD file, specify the CSDBKUP=DYNAMIC system initialization parameter. Also specify CSDRECOV=ALL and CSDFRLOG=*nn* to request forward recovery support.

## Removing BWO attributes

If you want to remove BWO attributes from your data sets, you must follow the correct procedure to avoid problems when taking subsequent back ups.

### Procedure

1. Close the VSAM data set either by shutting down CICS normally or issuing the command **CEMT SET FILE CLOSED**. Do not perform an immediate shutdown, as CICS does not close the files and the status of BWO does not reset. The BWO status of your data sets will not be correct when you restart CICS.
2. Alter the attributes for the data set to remove the BWO options. You can use the IDCAMS **ALTER NULLIFY BWO** command.
3. Restart CICS or reopen the data set.

## Systems administration

The systems administrator must decide which VSAM user data sets are eligible for BWO, and then set up the appropriate operating procedures for taking the BWO backup copies and for forward recovery.

These procedures should include:

- How to forward recover a data set by using the BWO backup copy, the forward recovery log, and the forward recovery utility to bring the data set to a point of consistency. Users must not have access to the file during the recovery process.
- How to forward recover a data set that may have been damaged while allocated to CICS. This operation may require backout of partially committed units of work during CICS emergency restart, after forward recovery has been done.

The procedures are simpler when using BWO than when not, because:

- Backups can be taken more frequently, so there are fewer forward recovery logs to manage. This also reduces the amount of processing that is required to forward recover the data set.
- The point from which forward recovery should start is recorded in the ICF catalog. The forward recovery utility uses this value to automate this part of the forward recovery process. This recovery point is saved with the backup copy and subsequently replaced in the ICF catalog when the backup copy is restored. For more information, see “Recovery point (non-RLS mode)” on page 137.
- During data set restore and forward recovery processing, CICS does not allow files to be opened for the same data set.

### Batch jobs

During the batch window between CICS sessions, it is possible for batch jobs to update a data set. Because batch jobs do not create forward recovery logs, any update that is made while a BWO backup is in progress, or after it has completed, would not be forward recoverable. Therefore, non-BWO backups should be taken, at least:

- At the start of the batch window so that, if a batch job fails, it can be restarted; and
- At the end of the batch window, for use with CICS forward recovery processing.

All update activity against the data set must be quiesced while the backups are taken, so that DFSMShsm can have exclusive control of the data set.

After an uncontrolled or immediate shutdown, further BWO backups might be taken by DFSMShsm, because the BWO status in the ICF catalog is not reset. These backups should be discarded; only the non-BWO backups taken at the end of the batch window should be used during forward recovery, together with the CICS forward recovery logs.

***Data set security***: CICS must have RACF ALTER authority for all data sets that are defined as BACKUPTYPE(DYNAMIC), because CICS needs to update the BWO attributes in the ICF catalog. The authority must apply either to the data set or to the ICF catalog in which the data set is cataloged. For information on defining RACF ALTER authority, see Authorizing access to CICS data sets.

## BWO processing

The information in the remainder of this section might be required by the system administrator to recover from error situations due to incorrect operating procedures or hardware failures.

The main data fields used by the BWO facility are:

- Attribute flags in the ICF catalog, to control BWO activity. The DFSMSdfp field dictionary name is VVRSMFLG. For more information about the attribute flags used in connection with BWO, see z/OS DFSMS Managing Catalogs.
- Recovery point in the ICF catalog. This point is the time from which the forward recovery utility must start applying log records. It is always before the time that the last backup was taken. It is recorded in local time format (0CYYDDDF HHMMSSTF) where:

  ```
  C   = century (0=1900, 1=2000, etc.)
  YY  = year
  DDD = day
  HH  = hours
  MM  = minutes
  SS  = seconds
  T   = tenths of a second
  F   = + sign
  ```

  The DFSMSdfp field dictionary name is VVRRDATA.
- The BACKUPTYPE attribute in the CICS file resource definition (for DFSMS 1.2), or the BWO option in the ICF catalog (for DFSMS 1.3). When CICS has determined the BWO option from one of these sources, it stores the value into the data set name block (DSNB) for the base cluster at the time the first file referencing the data set is opened.

The attribute flags and recovery point are managed by VSAM in the primary data VSAM volume record (VVR) of the base cluster, which is in the VSAM volume data set (VVDS). There is only one primary base cluster VVR for each VSAM sphere, which is why BWO eligibility is defined at the sphere level. For more information, see z/OS DFSMS Managing Catalogs.

BWO processing affects the following operations in a CICS system:

- File opening
- File closing
- Shutdown and restart
- Data set backup and restore
- Journaling and forward recovery logging
- Forward recovery

Each of these operations is discussed in the following sections.

## File opening

Different processing is done for each of the three cases when a file is opened for an update.

The following processing takes place:

- First file opened for update against a cluster
- Subsequent file opened for update against a cluster while the previous file is still open (that is, the update use count in the DSNB is not zero)
- Subsequent file opened for update against a cluster after all previous files have been closed (that is, the update use count in the DSNB is zero)

In all three cases, CICS issues warning message DFHFC5812 if BACKUPTYPE(DYNAMIC) is specified for a VSAM AIX data set that is being opened as a stand-alone base cluster data set. The AIX data set must default to the BACKUPTYPE already defined for the sphere.

Also, if the file-open operation fails during BWO processing, the ACB will be open. So CICS closes the ACB before indicating the file-open operation has failed. This affects CICS statistics.

If the file is opened for read-only, and the data set ICF catalog indicates that the data set is back-level, the file-open operation fails.

### Back-level data sets

In all cases, a file-open operation fails with error messages if the ICF catalog indicates that the data set is back-level. A back-level data set occurs as a result of the following:

- A data set has been restored from a backup copy, but not forward recovered.
- A data set has been forward recovered, but the forward recovery operation has not completed successfully.
- The ICF catalog indicates that a data set is corrupted.

**Note:** This check occurs irrespective of whether BACKUPTYPE(DYNAMIC) or BACKUPTYPE(STATIC) is specified.

**First file opened in non-RLS mode against a cluster:**

The following processing is done for the first file that is opened for update against a VSAM base cluster data set after a CICS cold start. (In this case, the update use count in the DSNB for the base cluster is always zero.)

CICS calls the DFSMSdfp IGWABWO callable service to inquire on the BWO attributes in the ICF catalog.

- If the file is defined with BACKUPTYPE(DYNAMIC), CICS calls IGWABWO to make the data set eligible for BWO and to set the recovery point to the current time. CICS also sets the BACKUPTYPE attribute in the DSNB to indicate eligibility for BWO.

  However, if the ICF catalog indicates that the data set is already eligible for BWO, IGWABWO just sets the recovery point to the current time. CICS issues a message, and you can discard any BWO backups already taken in a previous batch window.

- If the file was defined with BACKUPTYPE(STATIC) and the ICF catalog indicates that the data set is already ineligible for BWO, CICS sets the BACKUPTYPE attribute in the DSNB to indicate ineligibility for BWO.

  However, if the ICF catalog indicates that the data set is currently eligible for BWO, IGWABWO makes it ineligible for BWO and sets the recovery point to the current time. CICS issues a message, and you can discard any BWO backups already taken in a previous batch window.

If BWO support is requested and the appropriate level of DFSMSdfp (as described in "BWO requirements" on page 125) is not correctly installed on the processor where CICS is running, the first file-open operation fails with error message DFHFC5811. Subsequent file-open operations are allowed, but CICS issues an attention message.

CICS also issues an attention message (DFHFC5813) for the first file-open operation if the appropriate levels of DFSMShsm and DFSMSdss are not installed on the processor where CICS is running. Ensure that they are installed on the processor where the BWO backup is to be made.

**Subsequent files opened when use count is not zero:**

The following processing is done when a subsequent file is opened for update against a VSAM base cluster data set and the update use count in the DSNB for the base cluster is not zero.

The ICF catalog has already been validated and set by the first file-open operation, so CICS just checks the BACKUPTYPE attributes in the resource definition and the DSNB. If they are not consistent, the file-open operation fails with error messages. You must then either correct the resource definition, or REMOVE the DSNB after closing all files that are open against the base cluster data set.

**Subsequent files opened when use count is zero:**

The following processing is done when a subsequent file is opened for update against a VSAM base cluster data set and the update use count in the DSNB for the base cluster is zero.

This situation could exist in the following cases:

- After a warm or emergency restart of CICS, because the BACKUPTYPE attribute in the DSNB is cataloged in the global catalog and is restored at CICS restart
- When all files that are open against a base cluster are closed, and then one or more are reopened

CICS checks the BACKUPTYPE attributes in the FCT and the DSNB. If they are inconsistent, the file-open operation fails with error messages. Either correct the CEDA definition, or REMOVE the DSNB after closing all files that are open against the base cluster data set. If the BACKUPTYPE attributes are consistent, CICS uses the DFSMSdfp IGWABWO callable service to inquire on the BWO attributes in the ICF catalog.

- If the file was defined with BACKUPTYPE(DYNAMIC), IGWABWO makes the data set eligible for BWO and sets the recovery point to the current time.

  However, if the ICF catalog indicates that the data set is already eligible for BWO, IGWABWO resets the recovery point to the current time. CICS issues an attention message; you can discard any BWO backup copies already taken in a previous batch window.

- If the file was defined with BACKUPTYPE(STATIC) and the ICF catalog indicates that the data set is already ineligible for BWO, the ICF catalog is not updated.

  However, if the ICF catalog indicates that the data set is currently eligible for BWO, IGWABWO makes it ineligible for BWO and sets the recovery point to the current time. CICS issues an attention message; you should discard any BWO backup copies already taken in a previous batch window.

## File closing (non-RLS mode)

When the last file that is open for update is closed against a VSAM base cluster data set, CICS uses the DFSMSdfp IGWABWO callable service to update the ICF catalog to indicate that the data set is no longer eligible for BWO and to reset the recovery point to the current time.

If a VSAM split has occurred while a file was open, CICS calls IGWABWO at file-close time to update the ICF catalog to prevent further BWO backups. If DFSMShsm is currently taking a BWO backup, it will discard the backup at the end of the backup operation.

The BWO attributes indicating that a split has occurred and that the data set is eligible for BWO are restored when the next file is opened for update against the data set. This ensures that DFSMShsm takes the correct action if a split occurs during backup processing, which spans CICS updating a file (causing a VSAM split), the file being closed, and then the file being reopened.

When CICS is terminated by a normal shutdown, all CICS files are closed. The ICF catalog is updated to suppress BWO activity during the batch window between CICS sessions. After an uncontrolled or immediate shutdown, or if there is a failure during file closing, the data set remains open and the BWO flags are not reset. See “Shutdown and restart” on page 134.

### Restriction for VSAM upgrade set

In some circumstances, it might not be possible to take either BWO or non-BWO backups of a data set. The VSAM UPDATE ACB ENQs for a sphere might remain, even though there are no files open for update against this sphere. This could happen if a VSAM sphere contains an alternate index in the upgrade set and the following actions occur:

1. The sphere is opened for update via a VSAM path. This causes VSAM to open for update all upgrade clusters for this sphere.
2. A file is opened for read-only against this sphere.
3. The original VSAM path is closed.

The data set is now ineligible for BWO backups because CICS file control has reset the BWO attributes in the ICF catalog. But, until all open ACBs in the sphere are closed, VSAM will not close the internal ACBs that are open for update, and thus it is not possible to take non-BWO backups either.

To remedy this situation, either:

- Close all ACBs for the sphere, or
- Open for update against the base cluster data set a file that is defined with BACKUPTYPE(DYNAMIC).

## Shutdown and restart

The way CICS closes files is determined by whether the shutdown is controlled, immediate, or uncontrolled.

**Controlled shutdown:**

During a controlled shutdown, CICS closes all open files defined to CICS. This ensures that, for files that are open for update and eligible for BWO, the BWO attributes in the ICF catalog are set to a 'BWO disabled' state .

If a failure occurs during shutdown so that CICS is unable to close a file, CICS issues warning message DFHFC5804. In this case, check the BWO attributes and, if necessary, either use DFSMSdfp IGWABWO callable service to set the attributes, or discard any BWO backups that are taken in the batch window that follows the shutdown.

**Immediate or uncontrolled shutdown:**

During an immediate or uncontrolled shutdown, CICS does not close the files defined to CICS, and so the BWO attributes in the ICF catalog are not updated.

Use the DFSMSdfp IGWABWO callable service to set the attributes (see "An assembler program that calls DFSMS callable services" on page 139 for an example of how to do this). Do not run any batch jobs before the next CICS restart. If you do, for releases prior to DFSMS 1.2, discard any BWO backups that are taken in the batch window.

For DFSMS 1.2 onward, the controls in DFSMS allow DFSMSdss to detect a backup that is invalidated if CICS applications are shut down (normally or abnormally) and if batch programs are executed that update the data set while the BWO backup is in progress. This allows DFSMSdss to discard the backup, which prevents DFSMShsm from erroneously discarding the oldest valid backup from the inventory maintained by DFSMShsm.

**Restart:** At the next CICS restart, the following BWO-dependent actions can occur when a data set is opened for update:

- If the BWO attributes in the ICF catalog are set to the 'BWO enabled' state, CICS issues warning message DFHFC5808.
- If the file has been redefined as BACKUPTYPE(STATIC), and:
  - A cold start of CICS has been performed
  - The original base cluster DSNB has been discarded
  - The BWO attributes in the ICF catalog are set to the 'BWO enabled' state

  CICS issues warning message DFHFC5809.
- If the file has been redefined as BACKUPTYPE(STATIC), and:
  - CICS has been warm or emergency restarted
  - The original base cluster DSNB has been kept

  CICS fails the file-open operation with message DFHFC5807.

## Data set backup and restore

BWO backups are taken at the VSAM sphere level. You can use DFSMShsm or DFSMSdss to take the backup copy. You are recommended to use DFSMShsm because, without DFSMShsm installed, you must supply automatic class selection (ACS) routines to fulfil the SMS requirements for BWO support.

When you use DFSMShsm, you still use DFSMSdss as the data mover. You can specify this using the DFSMShsm SETSYS command:

```
SETSYS DATAMOVER(DSS)
```

The DFSMS processing at the start of backup is dependent on the DFSMS release level. For releases before DFSMS 1.2, DFSMSdss first checks the BWO attributes in the ICF catalog to see if the data set is eligible for BWO. If it is, the backup is made without attempting to obtain exclusive control and serialize updates to this data set.

For DFSMS 1.2 onward, DFSMSdss first tries to obtain exclusive control of the data set. If DFSMSdss succeeds, an enqueued form of backup takes place. If this serialization fails, DFSMSdss checks the BWO attributes in the ICF catalog to see if the data set is eligible for BWO. If it is, a BWO backup is attempted. If it is not eligible, the backup attempt fails.

This change will prevent DFSMS starting a BWO backup after CICS has abnormally terminated.

At the end of the BWO backup, DFDSS again checks the BWO attributes. If the data set is no longer eligible for BWO, the backup is discarded. Events that cause this situation are:

- File closing during BWO, which sets the 'BWO disabled' state
- Start of VSAM split, which sets the 'BWO enabled and VSAM split in progress' state
- End of VSAM split, which sets the 'BWO enabled/disabled and VSAM split occurred' state.

At the start of a backup, if the state is 'BWO enabled and VSAM split occurred', DFSMSdss resets the state to 'BWO enabled'. Then, if another VSAM split occurs, the backup will be discarded at the end of the backup operation.

**VSAM access method services:**

DFSMS access method services import and export operations do not support BWO in releases earlier than DFSMS 1.2. Access method services always serializes the data set before exporting it; and when the IMPORT function is used, the BWO attributes in the ICF catalog are not updated.

For DFSMS 1.2 onward, access method services supports the import and export of BWO attributes.

**Invalid state changes for BWO attributes:**

CICS, DFSMSdfp, DFSMSdss, and an SMSVSAM server can all update the BWO attributes in the ICF catalog.

To prevent errors, DFSMSdss fails a BWO backup if one of the following state changes is attempted during the backup:

- From 'BWO enabled and VSAM split in progress' to 'BWO enabled'. This state change could be attempted if:
  1. At the start of data set backup processing, a request is issued to change the 'BWO enabled and VSAM split occurred' state to the 'BWO enabled' state.

2. But then, before the 'BWO enabled' state is set, a VSAM split occurs and sets the 'BWO enabled and VSAM split in progress' state.

DFSMSdfp must now disallow the pending change to 'BWO enabled' (and DFSMSdss must fail the backup) because, if the split did not finish before the end of the backup, the invalid backup would not be discarded.

- From 'BWO disabled and VSAM split occurred' to 'BWO enabled'. This state change could be attempted if:
  1. At the start of data set backup processing, a request is issued to change the 'BWO enabled and VSAM split occurred' state to the 'BWO enabled' state.
  2. But then, before the 'BWO enabled' state is set, CICS closes the last file opened for update, and the data set becomes ineligible for BWO. CICS sets the 'BWO disabled and VSAM split occurred' state to ensure that the BWO backup is discarded and that no more BWO backups are taken.

  DFSMSdfp must now disallow the pending change to 'BWO enabled' (and DFSMSdss must fail the backup) to prevent the possibility of a BWO backup being taken during a subsequent batch window.

**Data set restore:**

When a BWO backup copy of a data set is restored, using DFSMShsm RECOVER or DFSMSdss RESTORE, the data set must be serialized to prevent any updates during the restore operation.

When the restore is complete, the BWO attributes in the ICF catalog are changed to a 'Backup restored by DFSMShsm' state. CICS cannot open the data set until forward recovery has been completed successfully.

DFSMSdss also resets the recovery point in the ICF catalog to the value it contained when the backup was made. This ensures that forward recovery starts at the correct point. This value should not be used for forward recovery of a non-BWO backup.

*Non-SMS managed storage:*

If a BWO backup is restored in storage that is not SMS-managed, the BWO attributes in the ICF catalog are lost. Thus forward recovery is not possible.

## Forward recovery logging

The forward recovery utility uses the forward recovery logs to recover a base cluster.

To do this, it must know:

- The data set to associate with each record on the logs
- The point from which to start recovery

**Data sets:**

Each data set after-image record on the log is associated with a file name.

However, there might be many files associated with the same data set; therefore, when a file is opened, the association between the file and the data set is recorded on the forward recovery log by a tie-up record. This information is also written to the log of logs. For non-BWO backups, the forward recovery utility uses this tie-up record to apply the log records to the correct data sets.

When a BWO is taken for a data set opened in RLS mode, DFSMSdss notifies each CICS region having an open ACB for the data set. On receipt of this notification, each CICS allows all units of work with updates for the data set to complete, and then they write the tie-up records to the forward recovery log and the log of logs, and replies to DFSMSdss.

For BWO backups, it is usually not necessary for the forward recovery utility to process a log from file-open time. Therefore, the tie-up records for all open files are written regularly on the log during activity-keypoint processing, and the time that they are written is recorded. To reduce the number of tie-up records if the activity keypoint frequency is high (such as for some large systems), CICS ensures that there is at least 30 minutes' separation between sets of tie-ups on the log.

**Recovery point (non-RLS mode):**

The recovery point is a time that can be converted to a position on a forward recovery log. Recovery of the data set requires only the records that are written after that position. Thus all previous records can be ignored by the forward recovery utility.

The recovery point is stored in the ICF catalog. It is initialized when the first file is opened for update against the data set, and updated during activity-keypoint processing and when the file is closed.

The recovery point is not the time of the current keypoint, as there might still be some uncommitted log records that have not been forced. Instead, it is the time of the start of the last keypoint that wrote a complete set of tie-up records and that completed earlier than the oldest uncommitted write to a forward recovery log.

**Note:**

1. Only one new recovery point is calculated during an activity keypoint. It is used for all data sets that are open for update and eligible for BWO. Thus a long-running task updating a data set that uses BWO will affect the amount of forward recovery needed for all data sets.
2. If you disable activity keypointing in your system (by specifying the AKPFREQ system initialization parameter as zero), BWO support is seriously affected because, after the file-open operation, no more tie-up records are written and the recovery point is not updated. So, forward recovery of a BWO data set must take place from the time that the data set was first opened for update.

## Forward recovery

CICS VSAM Recovery for z/OS fully supports BWO and the log of logs.

If you do not use CICS VSAM Recovery for z/OS, ensure that your forward recovery utility is able to:

- Recognize whether a backup was made with BWO or not. The DFSMShsm ARCXTRCT macro can be used to determine this.
- Use the BWO attributes and recovery point in the ICF catalog. It should use the DFSMSdfp IGWABWO callable service to do this. See "An assembler program that calls DFSMS callable services" on page 139 for a sample program.
- Recognize the additional tie-up records on the forward recovery logs and, optionally, recognize tie-up records on the log of logs. These are written so that the forward recovery utility can quickly find the correct position without having to scan the entire forward recovery log.
- Recognize after-images that have already been applied to the data set.

The forward recovery utility should ALLOCATE, with DISP=OLD, the data set that is to be recovered. This prevents other jobs accessing a back-level data set and ensures that data managers such as CICS are not still using the data set.

Before the data set is opened, the forward recovery utility should set the BWO attribute flags to the 'Forward recovery started but not ended' state. This prevents DFSMShsm taking BWO backups while forward recovery is in progress. To prevent CICS opening a back-level data set, the utility should perform this state change for all data sets in a system that supports BWO, even if some do not use BWO.

The forward recovery utility should use the BWO time stamp for the data set in the ICF catalog, set by DFSMSdss when the data set is restored, to determine the starting point in the forward recovery log to start the forward recovery.

If forward recovery completes successfully, the utility should set the BWO attributes to the 'BWO disabled' state before the data set is closed.

If forward recovery does not complete successfully, the utility should leave the BWO attributes in the 'Forward recovery started but not ended' state to ensure that CICS does not open a back-level data set.

If forward recovery does not complete successfully:

1. Determine and correct the reason for the failure
2. Delete the partially-recovered data set
3. Restore the backup copy
4. Reattempt forward recovery

**Note:** The EXEC CICS SET DSNAME RECOVERED system programmer command (or the CEMT equivalent) resets the BWO attributes in the ICF catalog to indicate 'BWO disabled'. If you use this command for a data set that has been restored but not forward recovered, and then you later open this data set, CICS is unaware that forward recovery has been overridden and CICS might access back-level data. However, in exceptional circumstances, it might be necessary to allow CICS to access back-level data. This command has been provided to allow this to happen.

Alternatively, if you use a VSAM forward recovery utility that does not update the BWO attributes during forward recovery, you may use these commands to reset the backup-restored-by-DFSMShsm state before subsequent CICS file control access.

## Recovering VSAM spheres with alternate indexes

Before you can forward recover a data set that was restored from a copy made using BWO, ensure that no alternate indexes are in the upgrade set. CICS VSAM Recovery for z/OS checks the upgrade set of data sets restored from BWO copies and issues a message if alternate indexes are found.

### About this task

To forward recover such a data set, after the restore, use the AMS ALTER or DELETE command to remove or delete the alternate indexes from the upgrade set. After forward recovery has completed successfully, you can re-create the upgrade set by rebuilding the alternate indexes using the access method services BLDINDX command.

## An assembler program that calls DFSMS callable services

```
*ASM XOPTS(CICS,NOEPILOG,SP)
*
*   A program that can be run as a CICS transaction to Read and Set
*   the BWO Indicators and BWO Recovery Point via DFSMS Callable
*   Services (IGWABWO).
*
*   Invoke the program via a CICS transaction as follows:
*
*   Rxxx 'data_set_name'
*   Sxxx 100 'data_set_name'
*
*   Where:
*    Rxxx and Sxxx are the names of the transactions that will invoke
*    this program. Specify Rxxx to read and Sxxx to set the BWO
*    attributes.
*    'data_set_name' is the fully-qualified name of your data set
*    100 is the value the BWO indicators will be set to.
*   The BWO Recovery Point time will be set to the current date and
*   time returned from the CICS ASKTIME function.
*
DFHEISTG DSECT
INDATA   DS   0CL53    Input data stream
*
* First character of tran id indicates transaction function
*
TRANFUNC DS   C        First char of tran id - S=SET R=READ
         DS   4C       Remainder of tran id and space
BWOC1    DS   C        First BWO indicator
BWOC2    DS   C        Second BWO indicator
BWOC3    DS   C        Third BWO indicator
         DS   C        Space
DSNAMES  DS   44C      Target data set name 1-44 chars
*
* 2 possible formats of input line, so overlay the definitions
*
         ORG  INDATA
         DS   5C       Tran id and space
DSNAMER  DS   44C      Target data set name 1-44 chars
         DS   4C       Filler
*
INLENGTH DS   H        Length of input data stream
*
* Parmlist for IGWABWO call
*
PARMLST  DS 10A
RETCODE  DS   F        Return code
REASON   DS   F        Reason
PROBDET  DS   D        Problem determination code
FUNC     DS   F        Function
READ     EQU  0           Read
SET      EQU  1           Set
DSNLEN   DS   F        Data set name length
DSN      DS   44C      Data set name
BWOFLAGS DS   03F      BWO indicator flags
         ORG  BWOFLAGS
BWOF1    DS   F           BWO indicator 1
BWOF2    DS   F           BWO indicator 2
BWOF3    DS   F           BWO indicator 3
BWOTIME  DS   D           BWO recovery point time
RESERVED DS   2D          Reserved
*
* Success message
*
SUCMSG   DS 0CL66         Define storage for success message
         DS 30C
```

```
DATEVAL  DS 8C            Date value from BWO recovery point
SUCMSG1  DS 8C            Message text
TIMEVAL  DS 8C            Time value from BWO recovery point
SUCMSG2  DS C             Message text
READMSG  DS 0CL11         If function = READ put out BWO flags
         DS 7C            Message text
BWOVAL1  DS C             BWO indicator 1
BWOVAL2  DS C             BWO indicator 2
BWOVAL3  DS C             BWO indicator 3
         DS C             Message text
*
DATETIME DS D             Current date and time value
*
RECOVPT  DS 0D            BWO recovery point
DTZERO   DS B                Date dword
DTCENTRY DS B
DTDATE   DS 5B
DTSIGN1  DS B
*
DTTIME   DS 6B               Time dword
DTTENTHS DS B
DTSIGN2  DS B
*
RECOVPTP DS 0D            Packed recovery point
DATEPACK DS F             Packed version of date
TIMEPACK DS F             Packed version of time
*
         DFHREGS
PROG     CSECT
PROG     AMODE 31
*
* Initialise INTO field for RECEIVE
*
         MVC  DSNAMER(48),BLANKS
         MVC  INLENGTH(2),INMAXLEN
*
         EXEC CICS RECEIVE INTO(INDATA) LENGTH(INLENGTH)
*
         CLI  TRANFUNC,C'S'       Set or Read call?
         BNE  PRGREAD
*
* Set up the parameters for a SET call
*
         SR   R4,R4
         LA   R4,SET(0)
         ST   R4,FUNC             Set function
         MVC  DSN(44),DSNAMES     Set data set name
         LH   R4,INLENGTH
         S    R4,PRELENS          Subtract tran id + space + BWO ind
         ST   R4,DSNLEN           Set data set name length
*
         EXEC CICS ASKTIME ABSTIME(DATETIME)
         EXEC CICS FORMATTIME ABSTIME(DATETIME) YYDDD(DTDATE)          *
              TIME(DTTIME)
*
         PACK KEYWORK(5),RECOVPT(9)   Packed date field
         MVC  DATEPACK(4),KEYWORK
         PACK KEYWORK(5),RECOVPT+8(9) Packed time field
         MVC  TIMEPACK(4),KEYWORK
         XC   RECOVPTP(1),RECOVPTP    Set century 0=1900, 1=2000
         OI   RECOVPTP+3,X'0F'        Set +ve sign for date
         OI   RECOVPTP+7,X'0F'        Set +ve sign for time
         MVC  BWOTIME(8),RECOVPTP     Set BWO recovery point time
*
         EXEC CICS SYNCPOINT
*
         MVC  BWOFLAGS(12),ZEROES
```

```
         LA   R4,1(0)
         CLI  BWOC1,C'0'
         BE   PRGBIT2
         ST   R4,BWOF1            Set BWO indicator 1 if required
PRGBIT2  DS   0H
         CLI  BWOC2,C'0'
         BE   PRGBIT3
         ST   R4,BWOF2            Set BWO indicator 2 if required
PRGBIT3  DS   0H
         CLI  BWOC3,C'0'
         BE   PRGCONT
         ST   R4,BWOF3            Set BWO indicator 3 if required
         B    PRGCONT
PRGREAD  DS   0H
         CLI  TRANFUNC,C'R'
         BNE  PRGABORT            If tran id not R or S then abort
*
* Set up the parameters for a read call
*
         SR   R4,R4
         LA   R4,READ(0)
         ST   R4,FUNC             Set function
         MVC  DSN(44),DSNAMER     Set data set name
         LH   R4,INLENGTH
         S    R4,PRELENR          Subtract tran id + space
         ST   R4,DSNLEN           Set data set name length
PRGCONT  DS   0H
*
* OK, our parameters are set up, so create the address list, and make
* the call
*
         LOAD EP=IGWABWO,ERRET=PRGABORT
         LR   R15,R0
         LA   R1,PARMLST          R1 -> parmlist
         LA   R4,RETCODE
         ST   R4,0(R1)            Pass addr of return code
         LA   R4,REASON
         ST   R4,4(R1)            Pass addr of reason code
         LA   R4,PROBDET
         ST   R4,8(R1)            Pass addr of problem determination
         LA   R4,FUNC
         ST   R4,12(R1)           Pass addr of function required
         LA   R4,DSNLEN
         ST   R4,16(R1)           Pass addr of data set name length
         LA   R4,DSN
         ST   R4,20(R1)           Pass addr of data set name
         LA   R4,SEL
         ST   R4,24(R1)           Pass addr of selection mask
         LA   R4,BWOFLAGS
         ST   R4,28(R1)           Pass addr of BWO flags
         LA   R4,BWOTIME
         ST   R4,32(R1)           Pass addr of BWO recovery point
         LA   R4,RESERVED
         ST   R4,36(R1)           Pass addr of reserved field
         BALR 14,15               Call IGWABWO
*
* Back from the call, check return code
*
         SR   R4,R4
         CL   R4,RETCODE          Check return code
         BNE  PRGABORT
*
* All OK, set up minimum success message, decide if we need more
*
         MVC  SUCMSG(38),SUCTXT       Set up message text
         MVC  SUCMSG1(8),SUCTXT1
```

```
         MVC  SUCMSG2(1),SUCTXT2
         UNPK KEYWORK(9),BWOTIME(5)   Make date printable
         TR   KEYWORK(8),HEXTAB-C'0'
         MVC  DATEVAL(8),KEYWORK
         UNPK KEYWORK(9),BWOTIME+4(5) Make time printable
         TR   KEYWORK(8),HEXTAB-C'0'
         MVC  TIMEVAL(8),KEYWORK
         CLI  TRANFUNC,C'S'           If READ then print BWO flags
         BNE  PRGREADO
*
* Got all the info we need, so put it out and exit
*
         EXEC CICS SEND TEXT FROM(SUCMSG) LENGTH(55) ERASE WAIT
*
         B    PRGEXIT
*
* It's a read so we also need the BWO flags for output
*
PRGREADO DS   0H
         MVC  READMSG(11),READTXT     Set up message text
         MVC  BWOVAL1,BWOF1+3
         OI   BWOVAL1,X'F0'           Set BWO indicator 1
         MVC  BWOVAL2,BWOF2+3
         OI   BWOVAL2,X'F0'           Set BWO indicator 2
         MVC  BWOVAL3,BWOF3+3
         OI   BWOVAL3,X'F0'           Set BWO indicator 3
*
* Now send the message
*
         EXEC CICS SEND TEXT FROM(SUCMSG) LENGTH(66) ERASE WAIT
*
PRGEXIT  DS 0H
         EXEC CICS RETURN
*
PRGABORT DS  0D
         EXEC CICS SEND TEXT FROM(FAILMSG) LENGTH(19) ERASE WAIT
*
         EXEC CICS RETURN
*
* Constant declarations
BLANKS   DC   48C' '
INMAXLEN DC   H'53'
ZEROES   DC   3F'0'
PRELENS  DC   F'9'
PRELENR  DC   F'5'
SUCTXT   DC   C'IGWABWO call completed Date = '
SUCTXT1  DC   C' Time = '
SUCTXT2  DC   C'.'
READTXT  DC   C' BWO =    .'
FAILMSG  DC   C'IGWABWO call failed'
KEYWORK  DC   CL9' '
HEXTAB   DC   C'0123456789ABCDEF'
*
* Constant for IGWABWO SELECT parameter
*
SEL      DC   F'3'        Interested in BWO flags & recov point
*             F'1'        Interested in BWO flags
*             F'2'        Interested in BWO recovery point
*             F'3'        Interested in BWO flags & recov point
         END PROG
```

# Disaster recovery

If your CICS system is normally available about 99 percent of the time, it would be wise to look at your disaster recovery plan. The same pressure that drives high availability drives the need for timely and current disaster recovery.

You must plan what level of disaster recovery you require for your CICS environment. If you are using DB2 or IMS, you can read more specific details related to database recovery in the following publications:

- *DB2 for z/OS Administration Guide*, for DB2 database recovery
- *IMS Operations Guide*, for IMS database recovery

See also the ITSC *Disaster Recovery Library: Planning Guide* for information that should help you set up a detailed disaster recovery plan if you use a combination of databases, such as DB2 and IMS.

## Why have a disaster recovery plan?

If your business cannot continue to function without CICS, you must have a disaster recovery plan.

To build a disaster recovery plan you must take into account a number of items unique to disaster recovery:

- What data is vital to my business?
- How long can the data be unavailable?
- How current does the data need to be?
- What is the cost of a disaster to my company?
- What is the cost of my disaster recovery plan?
- Is performance after a disaster a consideration?
- What type of disaster is possible, or even likely, and how long will it affect my system?

You might consider some, or all, of your CICS applications as vital to the operations of your business. If all applications are vital, you need to recover all the data that your CICS systems use. If only some of your applications are vital, you have to determine what data is associated with those applications.

The length of time between the disaster and recovery of your vital applications is a key factor. If your business cannot continue without access to your CICS data, your disaster recovery plan must take this into account.

The time-sensitive nature of your recovered data can be an overriding factor. If your vital application is a high volume, high change application, recovering week-old data may not be acceptable—even hour-old data may be unacceptable. You may need to recover right up to the point of the disaster.

The type of disaster from which you plan to recover can determine where your disaster recovery site is located. If you foresee only fire and water damage to your computer floor, a disaster recovery site in the building next door may be acceptable. If you are located in an area prone to hurricanes or earthquakes, for example, a disaster recovery site next door would be pointless.

When you are planning for disaster recovery, consider the cost of being unable to operate your business for a period of time. You have to consider the number of

lost transactions, and the future loss of business as your customers go elsewhere. Your disaster recovery solution should not be more expensive than the loss from the disaster, unless your business would fail as a result of the outage caused by a disaster.

What is the real cost of your disaster recovery plan? Keeping track of the total cost of your disaster recovery procedures allows you to look at available options and judge the benefits and cost of each.

Your disaster recovery plan should include some performance considerations once you have recovered. Unless your disaster site mirrors your production site, you should determine acceptable levels of throughput and transaction times while operating from the disaster recovery site. The length of time it takes to recover your primary site can also determine what your disaster recovery site has to support in the interim.

In summary, be aware that risk, speed of recovery, and completeness of recovery have to be balanced against cost.

## Disaster recovery testing

Testing is an essential part of disaster recovery planning. All too frequently, just creating a disaster recovery plan results in a false sense of security. If you don't test your disaster recovery plan, there's a risk that it won't work when you really need it.

Whenever possible, you should choose a remote site recovery strategy that you can test frequently. Testing your disaster recovery process has the following benefits:

- You know that your recovery plan works.
- You discover problems, mistakes, and errors, and can resolve them before you have to use the procedures.
- Your staff are educated in executing tests and managing disaster recovery situations.
- Your recovery plan becomes a living document.
- Members of your IT organization recognize the necessity of such a disaster recovery concept, and plan accordingly.
- Awareness of your disaster recovery strategy is increased.

After each test, use the detailed logs and schedules to identify any errors in your procedures, and eliminate them. Retest the changed procedures, and then incorporate them into your recovery plan. After changing the recovery plan, completely revise all existing disaster recovery documents.

Make frequent tests early in the implementation of your disaster recovery plan. Once you have removed the major problems, you can test less frequently. The frequency will depend on:

- The interval between major changes in your hardware and software
- How current you want to keep the recovery plan
- How critical and sensitive your business processes are: the more critical they are, the more frequently testing may be required.

## Six tiers of solutions for off-site recovery

One blueprint for recovery planning describes a scheme consisting of six tiers of off-site recoverability (tiers 1-6), with a seventh tier (tier 0) that relies on local recovery only, with no off-site backup.

The tiers cover a full range of recovery options, ranging from no data moved off-site to full off-site copies with no loss of data. The following figures and text describe them from a CICS perspective.

### Tier 0: no off-site data

Tier 0 is defined as having no requirements to save information off-site, establish a backup hardware platform, or develop a disaster recovery plan. Tier 0 is the no-cost disaster recovery solution.

Figure 16 summarizes the tier 0 solution.

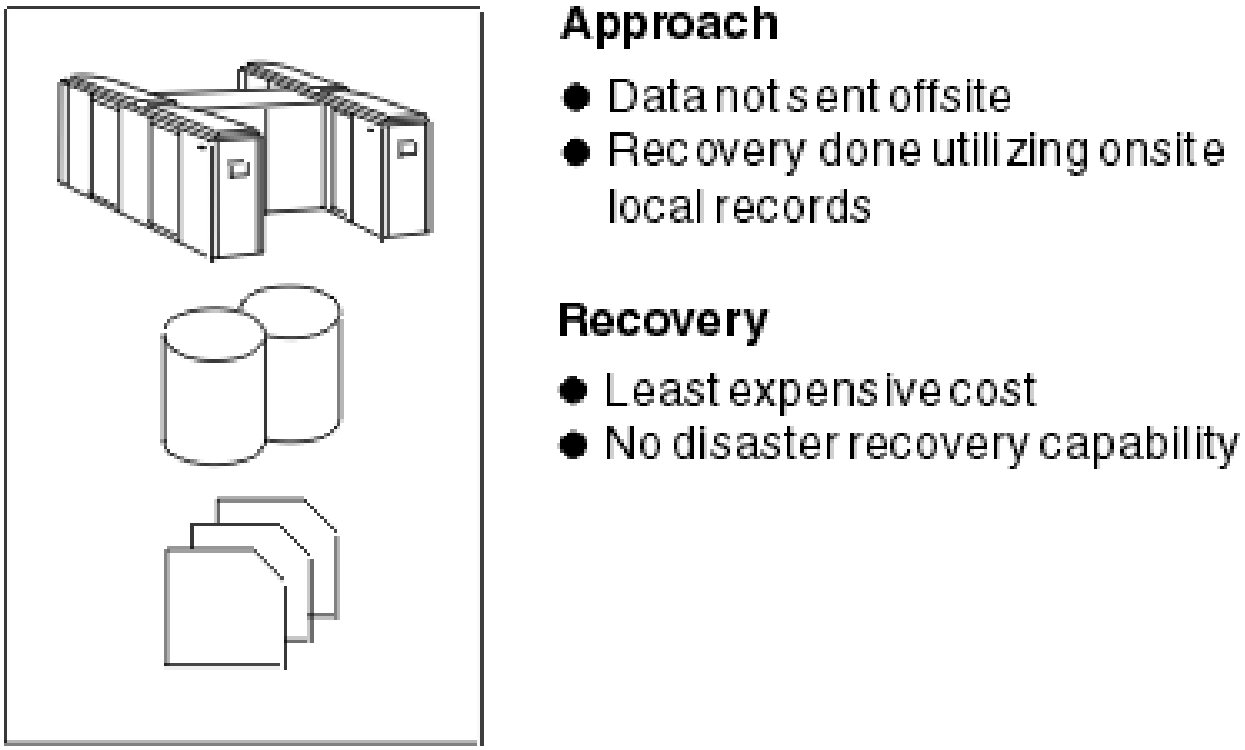



*Figure 16. Disaster recovery tier 0: no off-site backup*

Any disaster recovery capability would depend on recovering on-site local records. For most true disasters, such as fire or earthquake, you would not be able to recover your data or systems if you implemented a tier 0 solution.

### Tier 1 - physical removal

Tier 1 is defined as having a disaster recovery plan, required data set backups physically removed and transported to an off-site storage facility, and optionally, a backup site, but without the required hardware currently installed.

Figure 17 on page 146 summarizes the tier 1 solution.

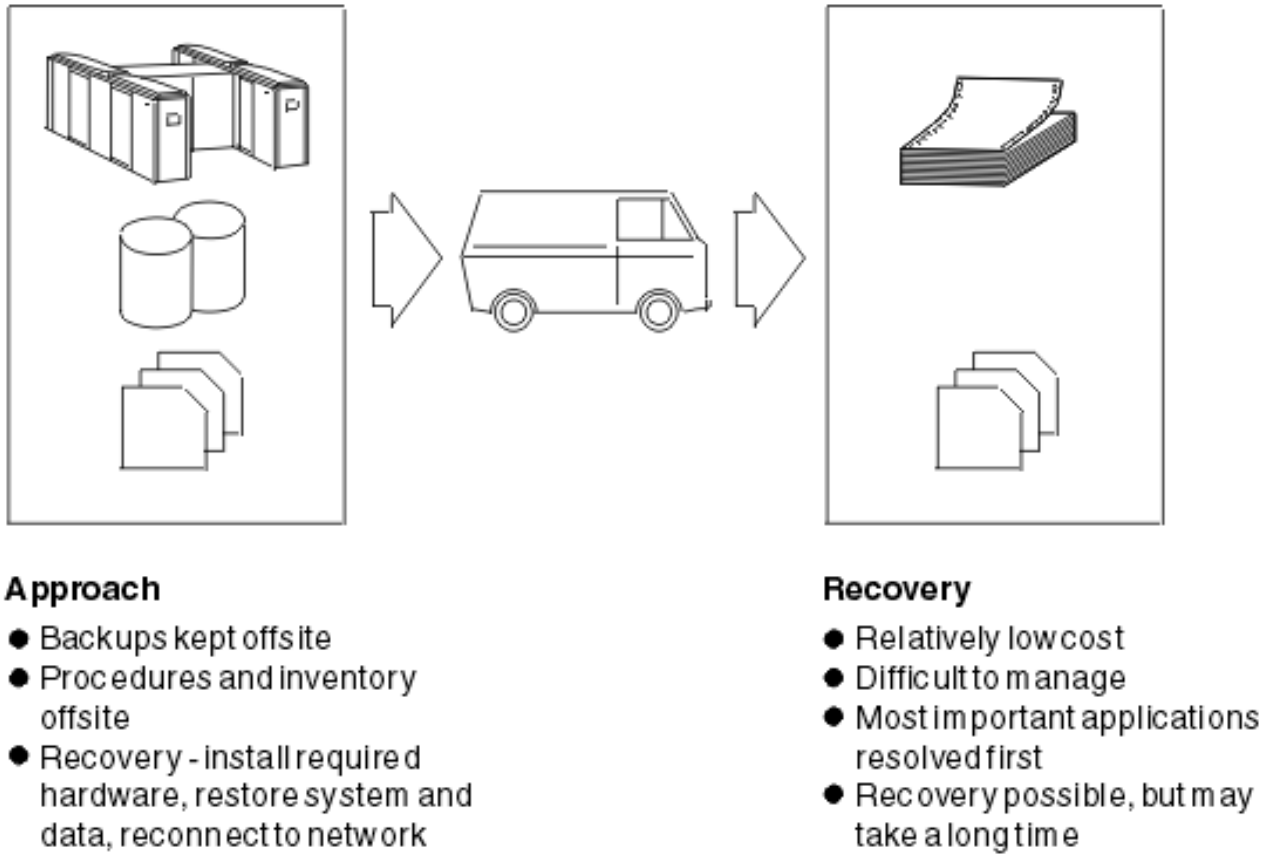



*Figure 17. Disaster recovery tier 1: physical removal*

Your disaster recovery plan has to include information to guide the staff responsible for recovering your system, from hardware requirements to day-to-day operations.

The backups required for off-site storage must be created periodically. After a disaster, your data can only be as up-to-date as the last backup—daily, weekly, monthly, or whatever period you chose—because your recovery action is to restore the backups at the recovery site (when you have one).

This method may not meet your requirements if you need your online systems to be continuously available.

- If you require data from two or more subsystems to be synchronized, for example, from DB2 and VSAM, you would have to stop updates to both, then copy both sets of data.
- Such subsystems would both be unavailable for update until the longest running copy is finished.
- If you require a point-in-time copy for all your data, your application may be unavailable for updates for a considerable time.

The major benefit of tier 1 is the low cost. The major costs are the storage site and transportation.

The drawbacks are:

- Setting up a computer floor, and obtaining the necessary hardware after a disaster can take a long time.

- Recovery is to the point in time of your last backups. You have no computer record, such as forward recovery logs, of any transactions that took place after these backups were taken.
- The process is difficult to manage.
- Your disaster recovery plan is difficult to test.

### Tier 1

Tier 1 provides a very basic level of disaster recovery. You will lose data in the disaster, perhaps a considerable amount. However, tier 1 allows you to recover and provide some form of service at low cost. You must assess whether the loss of data and the time taken to restore a service will prevent your company from continuing in business.

## Tier 2 - physical removal with hot site

Tier 2, like tier 1, provides a very basic level of disaster recovery. You will lose data in the disaster, perhaps a considerable amount.

However, tier 2 allows you to recover and provide some form of service at low cost and more rapidly than tier 1. You must assess whether the loss of data and the time taken to restore a service will prevent your company from continuing in business.

Figure 18 summarizes the tier 2 solution.

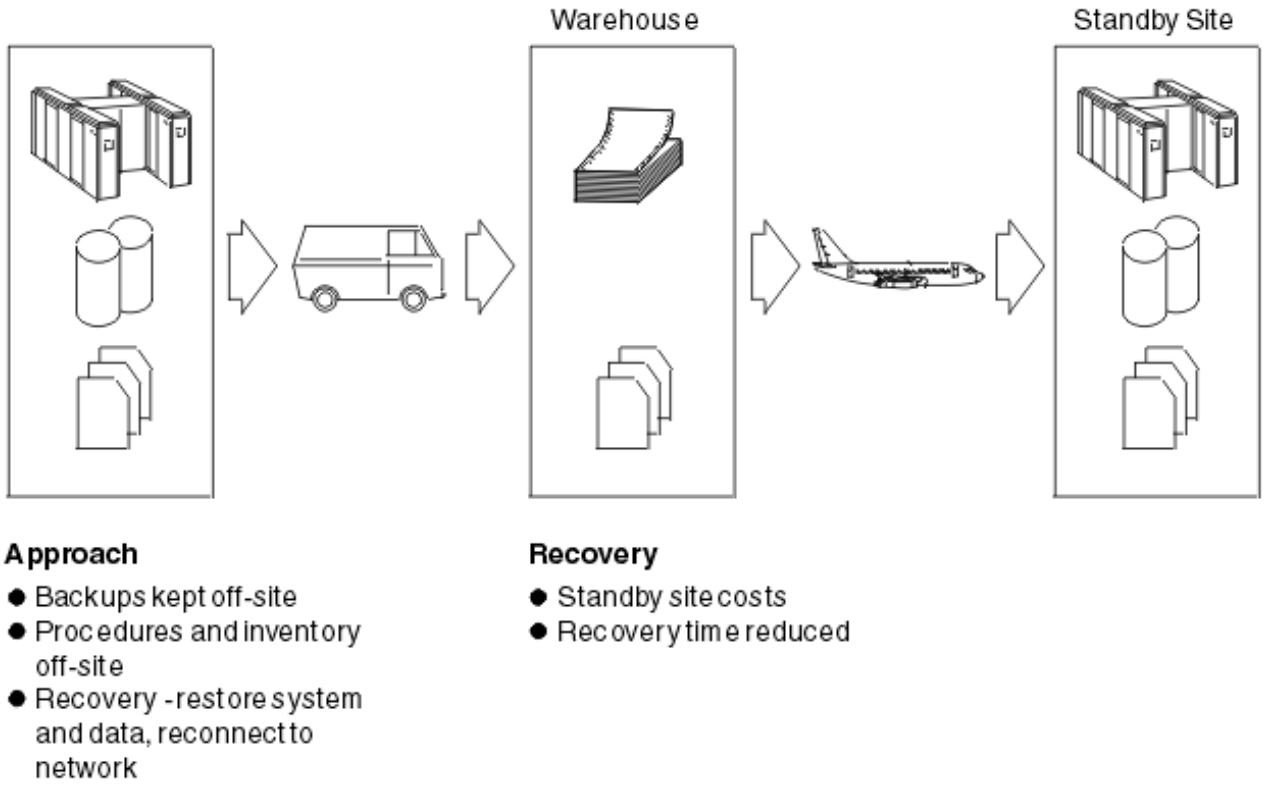



*Figure 18. Disaster recovery tier 2: physical removal to a 'hot' standby site*

Tier 2 is similar to tier 1. The difference in tier 2 is that a secondary site already has the necessary hardware installed, which can be made available to support the

vital applications of the primary site. The same process is used to backup and store the vital data; therefore the same availability issues exist at the primary site as for tier 1.

The benefits of tier 2 are the elimination of the time it takes to obtain and setup the hardware at the secondary site, and the ability to test your disaster recovery plan.

The drawback is the expense of providing, or contracting for, a 'hot' standby site.

### Tier 3 - electronic vaulting

Tier 3, like tiers 1 and 2, provides a basic level of disaster recovery. You will lose data in the disaster, perhaps a considerable amount of data.

The advantage of tier 3 is that you should be able to provide a service to your users quite rapidly. You must assess whether the loss of data will prevent your company from continuing in business.

Figure 19 summarizes the tier 3 solution.

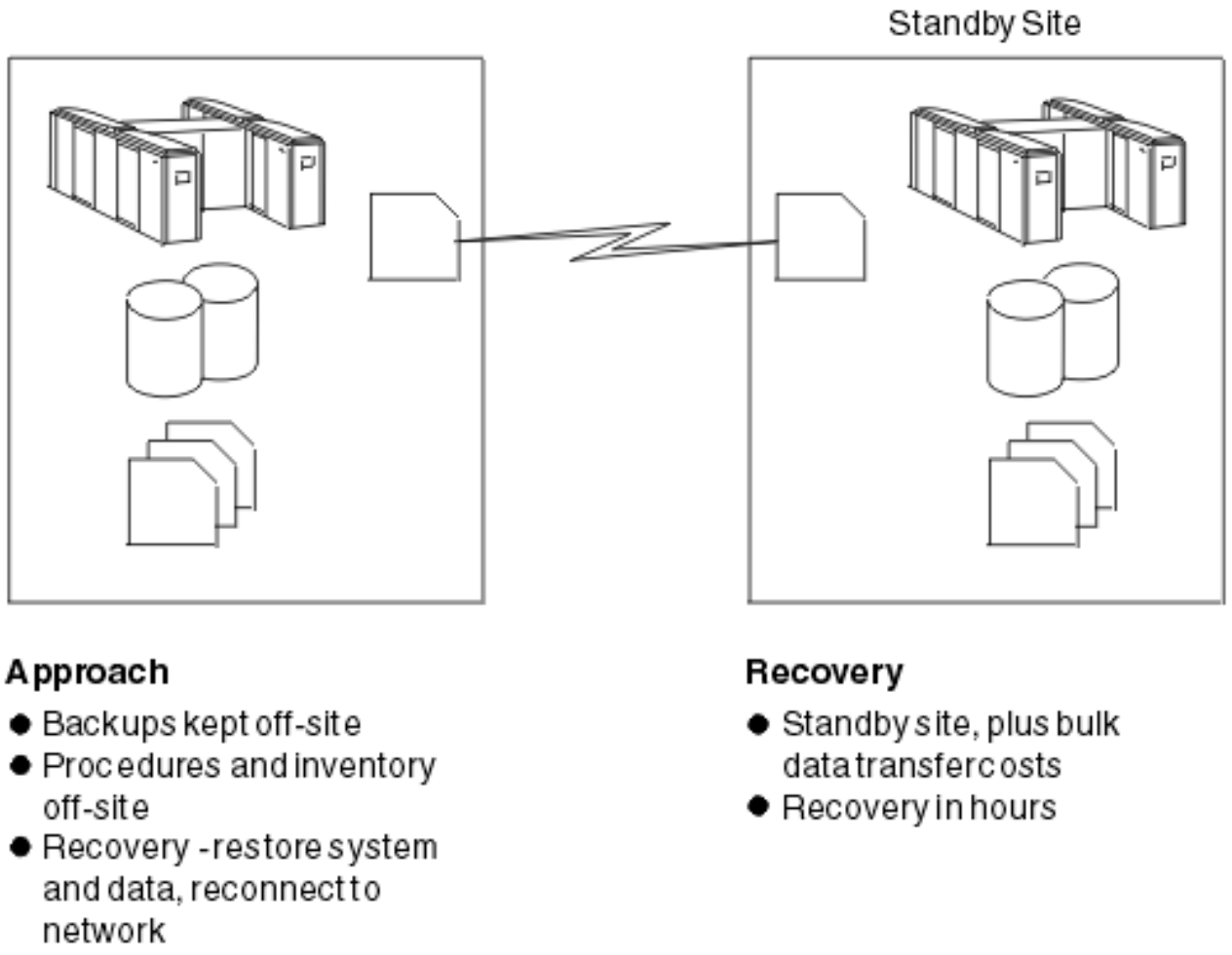



*Figure 19. Disaster recovery tier 3: electronic vaulting*

Tier 3 is similar to tier 2. The difference is that data is electronically transmitted to the hot site. This eliminates physical transportation of data and the off-site storage

warehouse. The same process is used to backup the data, so the same primary site availability issues exist in tier 3 as in tiers 1 and 2.

The benefits of tier 3 are:

- Faster recovery, as the data does not have to be retrieved from off-site and down-loaded.
- No need to ship the backups manually to a warehouse and store them.

The drawbacks are the cost of reserving the DASD at the hot standby site, and that you must have a link to the hot site, and the required software, to transfer the data.

Procedures and documentation still have to be available at the hot site, but this can be achieved electronically.

## Tier 0–3 solutions

Tiers 0 to 3 cover the disaster recovery plans of many CICS users. With the exception of tier 0, they employ the same basic design using a point-in-time copy of the necessary data. That data is then moved off-site to be used when required after a disaster.

Figure 20 summarizes the solutions for tiers 0 through 3, and shows the approximate time required for a recovery with each tier of solution.

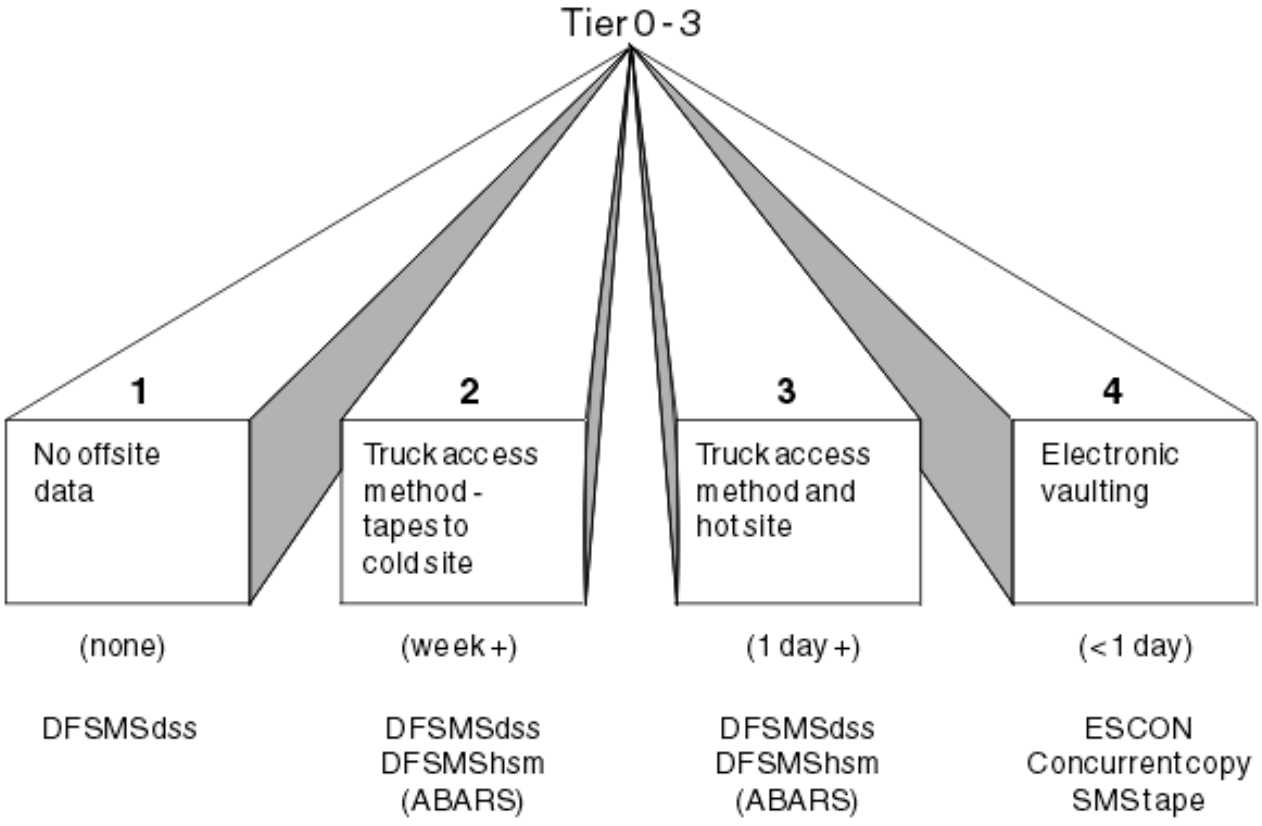



*Figure 20. Disaster recovery tier 0-3: summary of solutions*

The advantage of these methods is their low cost.

The disadvantages of these methods are:

- Recovery is slow, and it can take days or weeks to recover.
- Any recovery is incomplete, because any updates made after the point-in-time backup are lost.
- Disaster recovery is risky, and difficulties in testing your disaster recovery plan could lead to incomplete recovery.

### Tier 4 - active secondary site

Tier 4 provides a more advanced level of disaster recovery. You will lose data in the disaster, but only a few minutes- or hours-worth.

You must assess whether the loss of data will prevent your company from continuing in business, and what the cost of lost data will be.

Figure 21 summarizes the tier 4 solution.

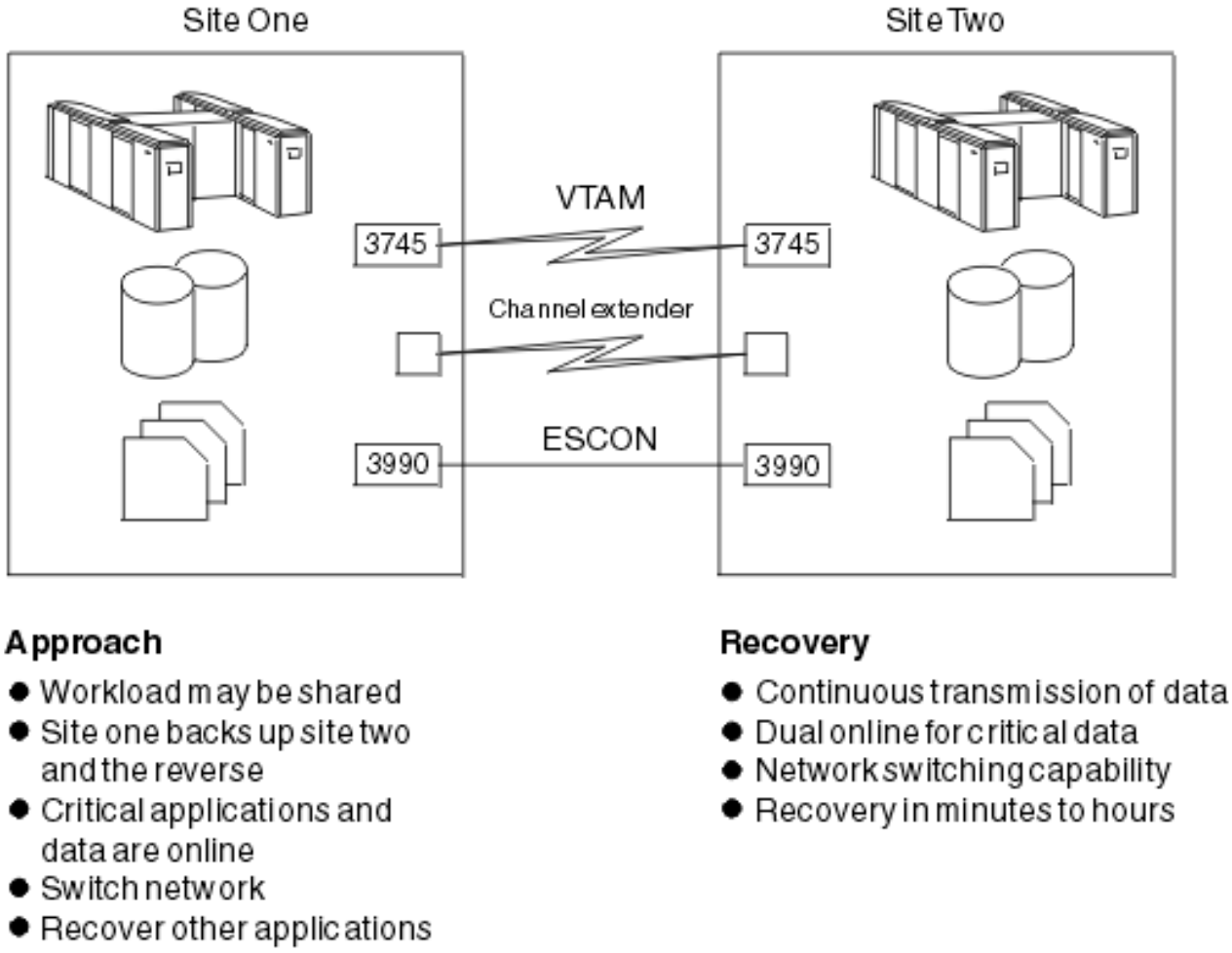



*Figure 21. Disaster recovery tier 4: active secondary site*

Tier 4 closes the gap between the point-in-time backups and current online processing recovery. Under a tier 4 recovery plan, site one acts as a backup to site two, and site two acts as a backup to site one.

Tier 4 duplicates the vital data of each system at the other's site. You must transmit image copies of data to the alternate site on a regular basis. You must also transmit CICS system logs and forward recovery logs, after they have been archived.

Similarly, you must transmit logs for IMS and DB2 subsystems. Your recovery action is to perform a forward recovery of the data at the alternate site. This allows recovery up to the point of the latest closed log for each subsystem.

You must also copy to the alternate site other vital data that is necessary to run your system. For example, you must copy your load libraries and JCL. You can do this on a regular basis, or when the libraries and JCL change.

The benefits of tier 4 are:

- Recovery is fast, with the required hardware and software already in place at the secondary site.
- Recovery is more complete than in the tier 1 to 3 solutions. You can recover all data to the end of the log for each of your data subsystems.
- Recovery risk is low, because you can easily test your procedures.

The drawbacks are:

- Recovery is more difficult to manage, as you have to ensure that all the logs and copies are transmitted to the other system.
- This solution introduces synchronization problems. Logs for different data subsystems are transmitted at different times. When you complete your recovery at the secondary site, you could find that your VSAM data is complete up to 30 minutes before the disaster, whereas your DB2 data is complete up to 15 minutes before the disaster. If your data must be synchronized, you may have to develop further procedures to resynchronize data in different subsystems.
- Cost is higher than for the tier 1 to 3 solutions, because you need dedicated hardware, software, and communication links.

## Tier 5 - two-site, two-phase commit

A tier 5 solution is appropriate for a custom-designed recovery plan with special applications. Because these applications must be designed to use this solution, it cannot be implemented at most CICS sites.

Figure 22 on page 152 summarizes the tier 5 solution.

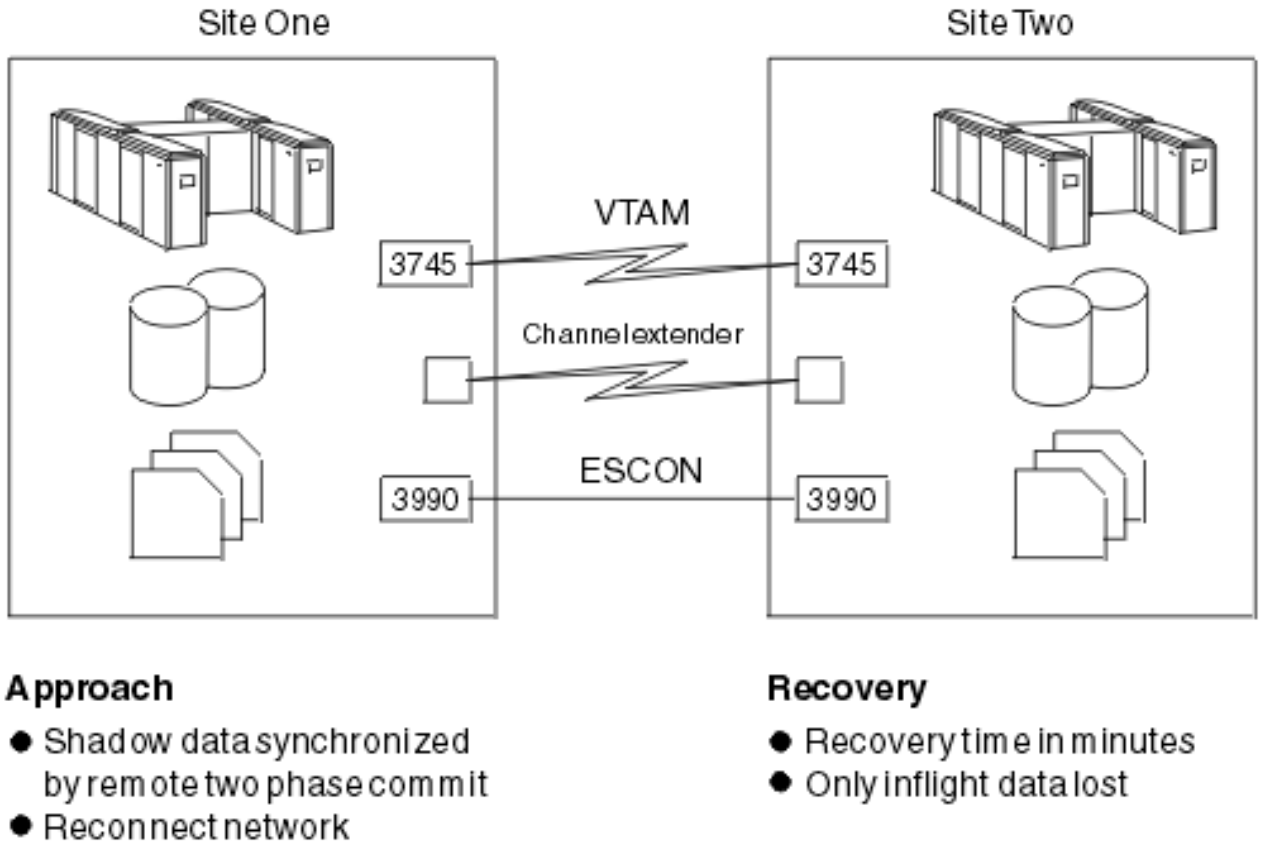



*Figure 22. Disaster recovery tier 5: two site, two-phase commit*

Tier 5, remote two-phase commit, is an application-based solution to provide high currency of data at a remote site. This requires partially or fully dedicated hardware at the remote site to keep the vital data in image format and to perform the two-phase commit. The vital data at the remote site and the primary site is updated or backed out as a single unit of work (UOW). This ensures that the only vital data lost would be from transactions that are in process when the disaster occurs.

Other data required to run your vital application has to be sent to the secondary site as well. For example, current load libraries and documentation has to be kept up-to-date on the secondary site.

The benefits of tier 5 are fast recovery using vital data that is current. The drawbacks are:

- The cost of maintaining and running two sites.
- The solution is application program dependent. You must write your applications so that they write data both locally and remotely, by transmitting data to a partner application at the remote site that also writes the data.
- This solution increases the time transactions take to respond. Your application program waits every time it needs to transmit data to the remote site.

### Tier 6 - minimal to zero data loss

Tier 6 provides a very complete level of disaster recovery. You must assess whether the cost of achieving this level of disaster recovery is justified for your company.

Figure 23 summarizes the tier 6 solution.

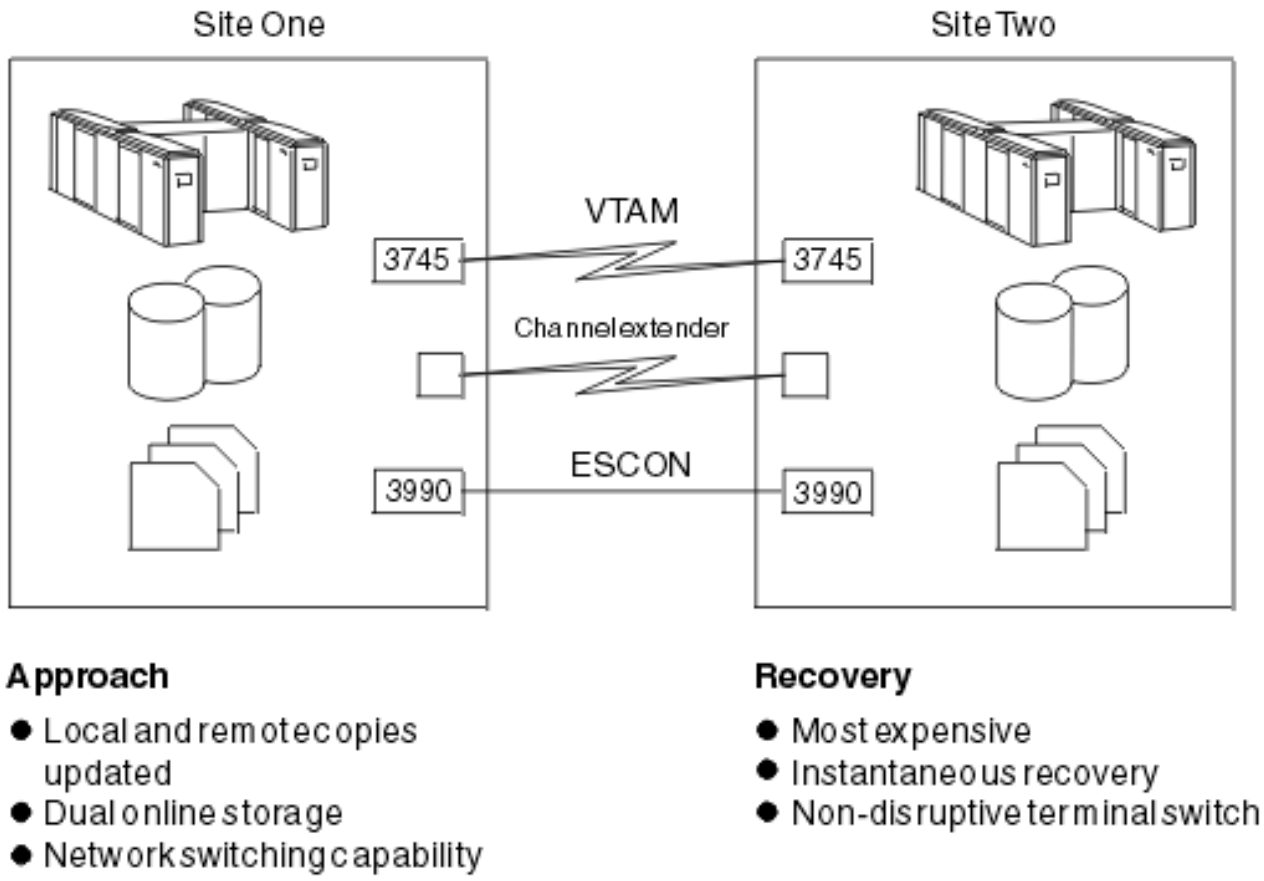



*Figure 23. Disaster recovery tier 6: minimal to zero data loss*

Tier 6, minimal to zero data loss, is the ultimate level of disaster recovery.

There are two tier 6 solutions, one hardware-based and the other software-based. For details of the hardware and software available for these solutions, see "Peer-to-peer remote copy (PPRC) and extended remote copy (XRC)" on page 155 (hardware) and "Remote Recovery Data Facility" on page 158 (software).

The hardware solution involves the use of IBM 3990-6 DASD controllers with remote and local copies of vital data. There are two flavors of the hardware solution: (1) peer-to-peer remote copy (PPRC), and (2) extended remote copy (XRC).

The software solution involves the use of Remote Recovery Data Facility (RRDF). RRDF applies to data sets managed by CICS file control and to the DB2, IMS, IDMS, CPCS, ADABAS, and SuperMICR database management systems, collecting real-time log and journal data from them. RRDF is supplied by E-Net Corporation and is available from IBM as part of the IBM Cooperative Software Program.

The benefits of tier 6 are:

- No loss of data.
- Recovery in a very short period of time.
- Emergency restart at remote site should be possible.

The drawbacks are the cost of running two sites and the communication overhead. If you are using the hardware solution based on 3990-6 controllers, you are limited in how far away your recovery site can be. If you use PPRC, updates are sent from the primary 3990-6 directly to the 3990-6 at your recovery site using enterprise systems connection (ESCON) links between the two 3990-6 devices. The 3990-6 devices can be up to 43 km (26.7 miles) apart subject to quotation.

If you use XRC, the 3990-6 devices at the primary and recovery sites can be attached to the XRC DFSMS/MVS host at up to 43 km (26.7 miles) using ESCON links (subject to quotation). If you use three sites, one for the primary 3990, one to support the XRC DFSMS/MVS host, and one for the recovery 3990, this allows a total of 86 km (53.4 miles) between the 3990s. If you use channel extenders with XRC, there is no limit on the distance between your primary and remote site.

For RRDF there is no limit to the distance between the primary and secondary sites.

## Tier 4–6 solutions

This summary shows the three tiers and the various tools for each that can help you reach your required level of disaster recovery.

Figure 24 summarizes the solutions for tiers 4 through 6, and shows the approximate time required for a recovery with each tier of solution.

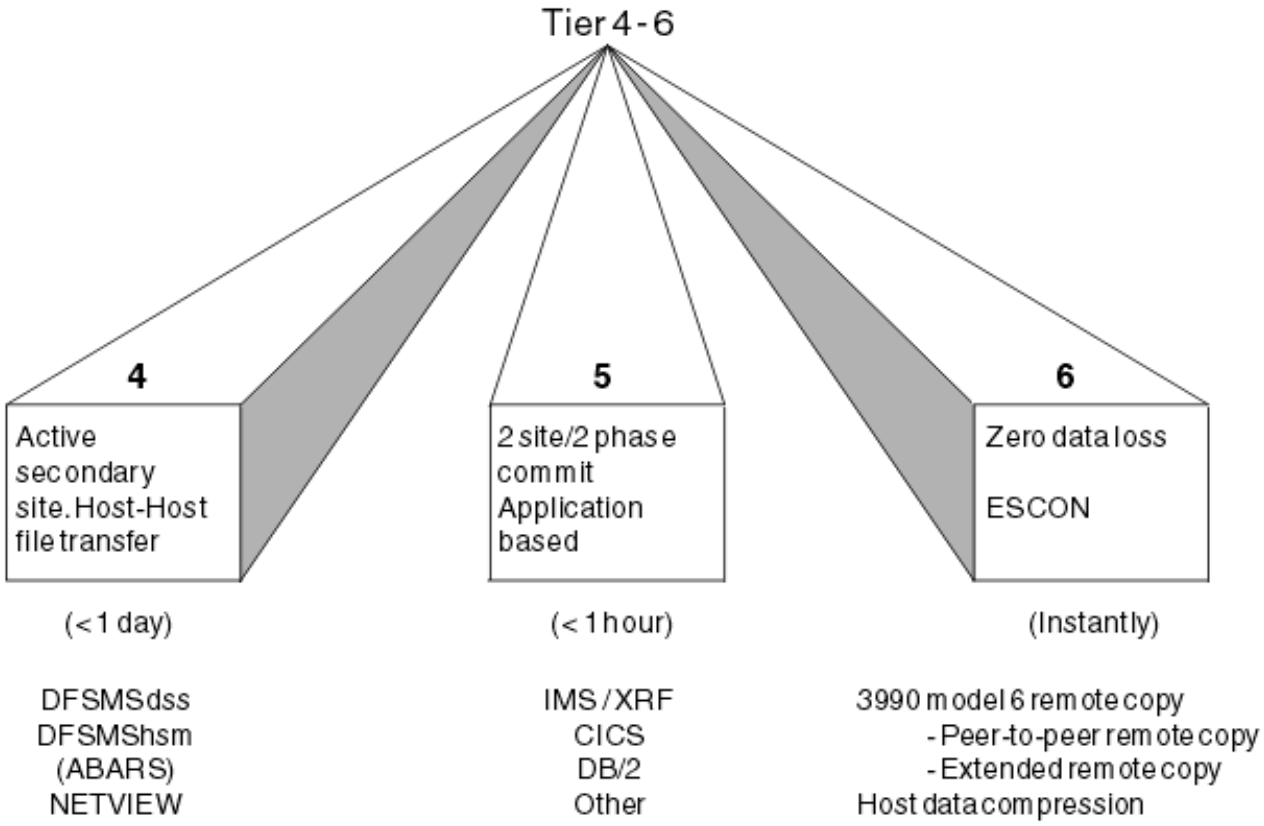



*Figure 24. Disaster Recovery Tier 4-6: Summary of Solutions*

Tier 4 relies on automation to send backups to the remote site. NetView provides the ability to schedule work in order to maintain recoverability at the remote site.

Tier 5 relies on the two-phase commit processing supported by various database products and your application program's use of these features. Tier 5 requires additional backup processing to ensure that vital data, other than databases, is copied to the remote system.

Tier 6 is divided into two sections: software solutions for specific access methods and database management systems, and hardware solutions for any data.

RRDF can provide very high currency and recoverability for a wide range of data. However, it does not cover all the data in which you may be interested. For example, RRDF does not support load module libraries.

The 3990-6 hardware solution is independent of the data being stored on the DASD. PPRC and XRC can be used for databases, CICS files, logs, and any other data sets that you need to ensure complete recovery on the remote system.

## Disaster recovery and high availability

This topic describes the tier 6 solutions for high availability and data currency when recovering from a disaster.

### Peer-to-peer remote copy (PPRC) and extended remote copy (XRC)

PPRC and XRC are both 3990-6 hardware solutions that provide data currency to secondary, remote volumes.

Updates made to secondary DASD are kept in time sequence. This ensures that updates are applied consistently across volumes. PPRC and XRC also ensure that updates are applied in time sequence across control units as well. This sequencing offers a very high degree of data integrity across volumes behind different control units.

Because PPRC and XRC are hardware solutions, they are application, and data, independent. The data can be DB2, VSAM, IMS, or any other type of data. All your vital data on DASD can be duplicated off-site. This reduces the complexity of recovery. These solutions can also make use of redundant array of independent disks (RAID) DASD to deliver the highest levels of data integrity and availability.

PPRC synchronously shadows updates from the primary to the secondary site. This ensures that no data is lost between the data committed on the primary and secondary DASD. The time taken for the synchronous write to the secondary unit has an impact on your application, increasing response time. This additional time (required for each write operation) is approximately equivalent to a DASD fastwrite operation. Because the implementation of PPRC is almost entirely in the 3990-6, you must provide enough capacity for cache and non-volatile storage (NVS) to ensure optimum performance.

XRC is an asynchronous implementation of remote copy. The application updates the primary data as usual, and XRC then passes the updates to the secondary site. The currency of the secondary site lags slightly behind the primary site because of updates in transit. As part of XRC data management, updates to the secondary site are performed in the same sequence as at the primary site. This ensures data integrity across controllers and devices. Because XRC does not wait for updates to be made at the secondary site, the application's performance is not directly affected. XRC uses cache and non-volatile storage, so you must provide enough capacity to ensure optimum performance.

In the event of a disaster, check the state of all secondary volumes to ensure data consistency against the shadowed log data sets. This ensures that the same sequence of updates is maintained on the secondary volumes as on the primary volumes up to the point of the disaster. Because PPRC and XRC do not require restores or forward recovery of data, your restart procedures on the secondary system may be the same as for a short-term outage at the primary site, such as a power outage.

When running with PPRC or XRC, the data you replicate along with the databases includes:

- CICS logs and forward recovery logs
- CICS system definition (CSD) data sets, SYSIN data sets, and load libraries
- Recovery control (RECON) and restart data set (RDS) for IMS
- IMS write-ahead data set (WADS) and IMS online log data set (OLDS)
- ACBLIB for IMS
- Boot-strap data set (BSDS), the catalog and the directory for DB2
- DB2 logs
- Any essential non-database volumes

CICS applications can use non-DASD storage for processing data. If your application depends on this type of data, be aware that PPRC and XRC do not handle it.

For more information on PPRC and XRC, see *Planning for IBM Remote Copy*, SG24-2595-00, and *DFSMS/MVS Remote Copy Administrator's Guide and Reference*.

**PPRC or XRC?:**

You need to choose between PPRC and XRC for transmitting data to your backup site. This topic compares the two methods to help you make your choice.

Choose PPRC as your remote copy facility if you:

- Require data currency at the secondary site
- Have your recovery site within ESCON distance
- Can accept some performance degradation
- Have a duplicate DASD configuration available at the remote site

The synchronous nature of PPRC ensures that, if you have a disaster at your main site, you lose only inflight transactions. The committed data recorded at the remote site is the same as that at the primary site.

**Use PPRC for high value transactions**

Consider PPRC if you deal with high value transactions, and data integrity in a disaster is more important to you than day-to-day performance. PPRC is more likely to be the solution for you if you characterize your business as being low volume, high value transactions; for example, a system supporting payments of thousands, or even millions, of dollars.

Choose XRC as your remote copy facility if you:

- Can accept that your data at the secondary site will be a few seconds behind the primary
- Have your secondary site outside ESCON distance

- Require high performance at the primary site

The asynchronous nature of XRC means that the remote site may have no knowledge of transactions that ran at the primary site, or does not know that they completed successfully. XRC ensures that the data recorded at the remote site is consistent (that is, it looks like a snapshot of the data at the primary site, but the snapshot may be several seconds old).

**Use XRC for high volume transactions**

Consider XRC if you deal with low value transactions, and data integrity in a disaster is less important to you than day-to-day performance. XRC is more likely to be the solution for you if you characterize your business as being high volume, low value transactions; for example, a system supporting a network of ATMs, where there is a high volume of transactions, but each transaction is typically less than 200 dollars in value.

**Other benefits of PPRC and XRC:**

PPRC or XRC may eliminate the need for disaster recovery backups to be taken at the primary site, or to be taken at all.

PPRC allows you to temporarily suspend the copying of updates to the secondary site. This allows you to suspend updates at the secondary site so that you can make image copies or backups of the data there. After the backups are complete, you can reestablish the pairing of data sets on the primary and secondary sites. Updates to the primary that have been recorded by the 3990-6 are applied to the secondary to resynchronize the pair.

XRC supports the running of concurrent copy sessions to its secondary volumes. This enables you to create a point-in-time copy of the data.

PPRC and XRC also allow you to migrate data to another or larger DASD of similar geometry, behind the same or different control units at the secondary site. This can be done for workload management or DASD maintenance, for example.

**Forward recovery:**

Whether you use PPRC or XRC, you have two fundamental choices. You can either pair volumes containing both the data and the log records or you can pair only the volumes containing the log records.

In the first case you should be able to perform an emergency restart of your systems, and restore service very rapidly. In the latter case you would need to use the log records, along with an image copy transmitted separately, to perform a forward recovery of your data, followed by an emergency restart.

Pairing the data volumes, as well as the log volumes, costs more because you have more data flowing between the sites, and therefore you need a greater bandwidth to support the flow. In theory you can restart much faster than if you have to perform a forward recovery. When deciding which to use, you must determine whether this method is significantly faster, and whether you think it is worth the additional costs.

## Remote Recovery Data Facility

The Remote Recovery Data Facility (RRDF), Version 2 Release 1, a product of the E-Net Corporation, minimizes data loss and service outage time in the event of a disaster by providing a real-time remote logging function.

Real-time remote logging provides data currency at the remote site, enabling the remote site to recover databases within seconds of the outage—typically in less than 1 second.

RRDF runs in its own address space. It provides programs that run in the CICS or database management system address space. These programs are invoked through standard interfaces—for example, at exit points associated with writing out log records.

The programs that run in the CICS or database management system address space use MVS cross-memory services to move log data to the RRDF address space. The RRDF address space maintains a virtual storage queue at the primary site for records awaiting transmission, with provision for spill files if communication between the primary and secondary sites is interrupted. Remote logging is only as effective as the currency of the data that is sent off-site. RRDF transports log stream data to a remote location in real-time, within seconds of the log operation at the primary site.

When the RRDF address space at the remote site receives the log data, it formats it into archived log data sets. Once data has been stored at the remote site, you can use it as needed to meet business requirements. The recovery process uses standard recovery utilities. For most data formats, first use the log data transmitted by RRDF in conjunction with a recent image copy of the data sets and databases that you have to recover. Then perform a forward recovery. If you are using DB2, you have the option of applying log records to the remote copies of your databases as RRDF receives the log records.

If you use DB2, you can use the optional RRDF log apply feature. With this feature you can maintain a logically consistent "shadow" copy of a DB2 database at the remote site. The RRDF log apply feature updates the shadow database at selected intervals, using log data transmitted from the primary site. Thus restart time is shortened because the work needed after a disaster is minimal. The currency of the data depends on the log data transmitted by RRDF and on how frequently you run the RRDF log apply feature. The RRDF log apply feature also enhances data availability, as you have read access to the shadow copy through a remote site DB2 subsystem. RRDF supports DB2 remote logging for all environments, including TSO, IMS, CICS, batch, and call attach.

At least two RRDF licenses are required to support the remote site recovery function, one for the primary site and one for the remote site. For details of RRDF support needed for the CICS Transaction Server, see "Remote Recovery Data Facility support" on page 160.

## Choosing between RRDF and 3990-6 solutions

### About this task

Table 4 on page 159 summarizes the characteristics of the products you can use to implement a tier 6 solution. You must decide which solution or solutions is most appropriate for your environment.

*Table 4. Selecting a tier 6 implementation*

| Characteristic | RRDF | 3990-6 |
|---|---|---|
| Data type supported | Various data sets[1] | Any on DASD |
| Database shadowing | Optional. Available for DB2 and IDMS only | Optional |
| Forward recovery required | Yes | Depends on implementation |
| Distance limitation | None | About 40 km for ESCON. Unlimited for XRC with channel extenders |
| **Note:** [1] Data sets managed by CICS file control and the DB2, IMS, IDMS, CPCS, ADABAS, and SuperMICR database management systems. | | |

### Disaster recovery personnel considerations

When planning for disaster recovery, you need to consider personnel issues.

You should ensure that a senior manager is designated as the disaster recovery manager. The recovery manager must make the final decision whether to switch to a remote site, or to try to rebuild the local system (this is especially true if you have adopted a solution that does not have a warm or hot standby site).

You must decide who will run the remote site, especially during the early hours of the disaster. If your recovery site is a long way from the primary site, many of your staff will be in the wrong place.

Finally, and to show the seriousness of disaster recovery, it is possible that some of your key staff may be severely injured and unable to take part in the recovery operation. Your plans need to identify backups for all your key staff.

### Returning to your primary site

One aspect of disaster recovery planning which can be overlooked is the need to include plans for returning operations from the recovery site back to the primary site (or to a new primary site if the original primary site cannot be used again).

#### About this task

Build the return to normal operations into your plan. The worst possible time to create a plan for moving back to your primary site is after a disaster. You will probably be far too busy to spend time building a plan. As a result, the return to your primary site may be delayed, and may not work properly.

## Disaster recovery facilities

This section looks at the various alternatives for achieving disaster recovery with CICS and also looks at utilities that can aid the process.

### MVS system logger recovery support

The MVS system logger provides support that enables a recovery resource manager to be associated with a log stream (that is, a local recovery resource manager operating on behalf of a remote site).

The name of the recovery resource manager is specified when a new log stream definition is created or an existing log stream definition is updated. When a recovery resource manager connects to the log stream, through the IXGCONN service, it requests that a resource manager-owned exit be given control when

specified events occur. When such an event occurs, the MVS system logger invokes the exit specified by the resource manager and passes it details of the event. It is then the responsibility of the recovery resource manager to transmit log records to a remote site.

The remote site needs to be able to import log streams transmitted by the recovery resource manager; this too is provided by MVS system logger services. Importation is the process whereby log blocks resident in one log stream (the source log stream) are created in (or copied into) another log stream (the target log stream) while maintaining (in the target log stream) the same MVS system logger-assigned log block id and GMT time stamp that is assigned to the source log stream. The result of a log stream import is a copy of the source log stream in the target log stream.

### CICS VSAM Recovery QSAM copy

CICS VSAM Recovery (CICS VR) provides a QSAM copy function that can copy MVS log streams to a QSAM data set.

Copies of the QSAM data can be sent either electronically or physically to the remote site. On arrival at the remote site, you can use the MVS system logger import services to put the log records into an MVS system logger log stream. Alternatively, you can use CICS VR to perform forward recovery of a data set using the QSAM data directly.

### Remote Recovery Data Facility support

The Remote Recovery Data Facility (RRDF) product from the E-Net Corporation supports the CICS log manager.

RRDF Version 2 Release 1 uses the disaster recovery services (for export and import of log streams) provided by the MVS system logger. RRDF connects to a log stream at the local site where the resource manager exit is specified, to register its interest. The recovery manager is given control whenever writes or deletes occur. Typically, writes are intercepted for transmission to the remote site. Delete requests are intercepted to prevent CICS from deleting system log records before RRDF has sent them to the remote site. RRDF at the remote site receives the transmitted log records, establishes an import connection to a log stream, and imports the log records.

### CICS VR shadowing

CICS VR provides a data shadowing facility. Shadowing helps to reduce recovery time by applying forward recovery logs periodically at the remote site. See CICS VR documentation for a complete explanation.

## CICS emergency restart considerations

It is important to consider the differences between CICS Transaction Server and earlier releases of CICS when planning for off-site recovery.

### Indoubt and backout failure support

Support during emergency restarts for units-of-work that failed indoubt or failed during backout is provided by the CICS recovery manager.

This support is available at the remote site only if the system log is transmitted and the CICS regions at the remote site are running under CICS Transaction Server.

It is possible for system log records to be transmitted to the remote site for units-of-work that subsequently become indoubt-failed or backout-failed. The log

records for failed units of work could be moved to the secondary log stream at the local site. However, resource managers such as RRDF are aware of such movements and act accordingly.

### Remote site recovery for RLS-mode data sets

CICS provides support for remote site recovery where VSAM data sets are used in RLS mode at the primary site. Using this RLS support for remote recovery, you can switch over to the remote site without suffering indeterminate or unreported loss of data integrity.

If a disaster occurs at the primary site, your disaster recovery procedures should include recovery of VSAM data sets at the designated remote recovery site. You can then emergency restart the CICS regions at the remote site so that they can backout any uncommitted data. Special support is needed for RLS because record locks, which were protecting uncommitted data from being updated by other transactions at the primary site, are not present at the remote site. You invoke CICS RLS support for off-site recovery using the OFFSITE system initialization parameter.

The OFFSITE system initialization parameter protects all RLS data sets until all emergency restart recovery is complete. You can specify this OFFSITE system initialization parameter at run-time only—it cannot be specified and assembled in the SIT—and it is valid only when START=AUTO is specified. You specify OFFSITE=YES when restarting CICS regions at a remote site when recovery involves VSAM data sets opened in RLS mode.

When you specify OFFSITE=YES, file control performs RLS off-site recovery processing, under which file control does not allow any new RLS access during startup. With RLS recovery in operation during an emergency restart, CICS does not allow any data sets to be accessed in RLS mode until:

- CICS has completed all outstanding RLS recovery work.
- CICS file control has issued a message requesting a "GO" response from an operator when all CICS regions have completed RLS recovery processing.
- An operator has replied to the message.

  Operators should reply to the message only when all the CICS regions being restarted with OFFSITE=YES have issued the message, indicating that they have all completed their RLS recovery.

A CICS-supplied sample NetView EXEC, DFH$OFAR, is provided to automate the detection of, and reply to, the WTOR console messages. For more information, see the prolog in the source member of the CICSTS54.CICSSDFHSAMP. library.

## Final summary

Your disaster recovery plan will be truly tested only at a very difficult time for your business—during a disaster. Careful planning and thorough testing may mean the difference between a temporary inconvenience and going out of business.

The goal of your disaster recovery plan is to get your CICS systems back online. The currency of the data and the time it takes to get back online is a function of which disaster recovery tier you use. Unless legal requirements for disaster recovery dictate the type of disaster recovery you must have, the choice of tiers is usually a business decision.

Making your disaster recovery work requires a good plan, up-to-date documentation, and regular testing.

# Notices

This information was developed for products and services offered in the U.S.A. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

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 rights 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 grant you any license to these patents. You can send license inquiries, in writing, to:

*IBM Director of Licensing*
*IBM Corporation*
*North Castle Drive, MD-NC119*
*Armonk, NY 10504-1785*
*United States of America*

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

*Intellectual Property Licensing*
*Legal and Intellectual Property Law*
*IBM Japan Ltd.*
*19-21, Nihonbashi-Hakozakicho, Chuo-ku*
*Tokyo 103-8510, Japan*

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 jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore this statement may not apply to you.

This information 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.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

© Copyright IBM Corp. 1989, 2017

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who want 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 Director of Licensing*
*IBM Corporation*
*North Castle Drive, MD-NC119*
*Armonk, NY 10504-1785*
*US*

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.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

## Programming interface information

CICS supplies some documentation that can be considered to be Programming Interfaces, and some documentation that cannot be considered to be a Programming Interface.

Programming Interfaces that allow the customer to write programs to obtain the services of CICS Transaction Server for z/OS, Version 5 Release 4 are included in the following sections of the online product documentation:

- Developing applications
- Developing system programs
- RACF security overview
- Developing for external interfaces
- Reference: application developmenth
- Reference: system programming
- Reference: connectivity

Information that is NOT intended to be used as a Programming Interface of CICS Transaction Server for z/OS, Version 5 Release 4 , but that might be misconstrued as Programming Interfaces, is included in the following sections of the online product documentation:

- Troubleshooting and support
- Reference: diagnostics

If you access the CICS documentation in manuals in PDF format, Programming Interfaces that allow the customer to write programs to obtain the services of CICS Transaction Server for z/OS, Version 5 Release 4 are included in the following manuals:

- Application Programming Guide and Application Programming Reference
- Business Transaction Services
- Customization Guide
- C++ OO Class Libraries
- Debugging Tools Interfaces Reference
- Distributed Transaction Programming Guide
- External Interfaces Guide
- Front End Programming Interface Guide
- IMS Database Control Guide
- Installation Guide
- Security Guide
- Supplied Transactions
- CICSPlex SM Managing Workloads
- CICSPlex SM Managing Resource Usage
- CICSPlex SM Application Programming Guide and Application Programming Reference
- Java Applications in CICS

If you access the CICS documentation in manuals in PDF format, information that is NOT intended to be used as a Programming Interface of CICS Transaction Server for z/OS, Version 5 Release 4 , but that might be misconstrued as Programming Interfaces, is included in the following manuals:

- Data Areas
- Diagnosis Reference
- Problem Determination Guide
- CICSPlex SM Problem Determination Guide

## Trademarks

IBM, the IBM logo, and ibm.com® are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at Copyright and trademark information at www.ibm.com/legal/copytrade.shtml.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

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

UNIX is a registered trademark of The Open Group in the United States and other countries.

## Terms and conditions for product documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

**Applicability**
These terms and conditions are in addition to any terms of use for the IBM website.

**Personal use**
You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM.

**Commercial use**
You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

**Rights** Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

## IBM online privacy statement

IBM Software products, including software as a service solutions, (“Software Offerings”) may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering’s use of cookies is set forth below:

**For the CICSPlex SM Web User Interface (main interface):**
Depending upon the configurations deployed, this Software Offering may use session and persistent cookies that collect each user’s user name and other personally identifiable information for purposes of session management, authentication, enhanced user usability, or other usage tracking or functional purposes. These cookies cannot be disabled.

**For the CICSPlex SM Web User Interface (data interface):**
Depending upon the configurations deployed, this Software Offering may use session cookies that collect each user's user name and other personally identifiable information for purposes of session management, authentication, or other usage tracking or functional purposes. These cookies cannot be disabled.

**For the CICSPlex SM Web User Interface ("hello world" page):**
Depending upon the configurations deployed, this Software Offering may use session cookies that collect no personally identifiable information. These cookies cannot be disabled.

**For CICS Explorer:**
Depending upon the configurations deployed, this Software Offering may use session and persistent preferences that collect each user’s user name and password, for purposes of session management, authentication, and single sign-on configuration. These preferences cannot be disabled, although storing a user's password on disk in encrypted form can only be enabled by the user's explicit action to check a check box during sign-on.

If the configurations deployed for this Software Offering provide you, as customer, the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, see IBM Privacy Policy and IBM Online Privacy Statement, the section entitled “Cookies, Web Beacons and Other Technologies” and the IBM Software Products and Software-as-a-Service Privacy Statement.

# Index

## Numerics


3270 interface
  application debugging profile manager 39


## A


abbreviations for keywords 8
abends
  cause of immediate shutdown 66
  CICS system 66
  during immediate shutdown 69
ACID properties, of a transaction 84
activating debugging profiles
  with the 3270 interface 43
  with the web interface 25
activity keypoints
  description 86
AID (automatic initiate descriptor)
  CICS system 68
AIRDELAY 102
application debugging profile manager 18
  3270 interface 39
    activating debugging profiles 43
    changing debugging profiles 44
    copying debugging profiles 45
    create Java debugging profile screen 51
    create LE debugging profile screen 47
    creating debugging profiles 42
    deleting debugging profiles 45
    inactivating debugging profiles 44
    list debugging profiles screen 39
    set LE debugging display device screen 57
    setting the display device 56
    view Java debugging profile screen 54
    view LE debugging profile screen 54
    viewing debugging profiles 44
  web interface 19
    activating debugging profiles 25
    changing debugging profiles 27
    configuring 19
    copying debugging profiles 27
    Create Java profile page 32
    Create LE profile page 29
    creating debugging profiles 24
    deleting debugging profiles 28
    inactivating debugging profiles 26
    List profiles page 20
    Set LE display device page 38
    setting the display device 36
    View Java profile page 36
    View LE profile page 35
application debugging profile manager *(continued)*
  web interface *(continued)*
    viewing debugging profiles 26
ARM
  de-registering 119
  unregister 119
atomic unit of work 84
autoinstall for terminals
  warm start 63
autojournals 87
automated operations 2
automatic journaling 87
automatic restart manager 118


## B


backup while open (BWO)
  attribute flags in ICF catalog 130
  CICS processing 130
  eligible data sets 125
  how to define files 127
  introduction 124
  main data fields used 130
  other products required 125
  systems administration 129
BACKUPTYPE attribute 127
batch jobs
  effect on BWO 129
BMS (basic mapping support)
  warm start 63
bundle recovery 116
BWO
  removing 129


## C


CADP transaction 39
canceling a transaction 7
catalogs
  failure 93
  global catalog contents 94
  use of in normal shutdown 89
CAVM (CICS availability manager)
  signoff abnormal at system 68
  signoff abnormal at system termination 69
CEMT transaction
  PERFORM SHUTDOWN 66
  PERFORM SHUTDOWN IMMEDIATE 66
chaining a message
  partitions 12
changing debugging profiles
  with the 3270 interface 44
  with the web interface 27
CICS system quiesce processing
  automatic initiate descriptor (AID) 68
CICS system quiesce processing *(continued)*
  CLSDST requests for z/OS Communications Server terminals 68
  file control 68
  first stage 67
  first stage PLT programs 67
  interregion communication 68
  operator notification 67
  QUIESCE_DOMAIN calls 67
  second stage 67
  second stage PLT programs 67
  signoff abnormal from CAVM 68
  subsystem interface 67
  terminal control 68
  termination task priority change 67
  transaction list table (XLT) 67
  warm keypointing 68
  warm-start-possible indicator 68
CICS system shutdow
  transaction routing 69
CICS system shutdown
  immediate 66, 68
  normal 66
  processing 67
  types 66
  uncontrolled 67
CICS system termination
  first stage processing 68
  resource managers 69
  second stage processing 68
  signoff abnormal from CAVM 69
  subsystem interface 69
  terminal control 69
  termination statistics 69
CLEAR key
  when invoking transaction 7
  when used with partitions 12
CLEAR PARTITION key 12
CLIST, TSO command list 2
cold start 103
  DL/I 60
  file control 60
  LIBRARY definitions 60
  MODEL definitions 60
  process 60
  PROFILE definitions 60
  program control 60
  PROGRAM definitions 60
  resource definition 61
  task control 60
  terminal control 60
  TERMINAL definitions 60
  TRANSACTION definitions 60
  TYPETERM definitions 60
communicating with CICS 6
console as a CICS terminal 14
console device
  entering transactions 2


© Copyright IBM Corp. 1989, 2017

console devices
using TSO command lists 2
console message-formatting 3
console support
communicating with CICS 1
console support, multiple 14
controlled shutdown
warm keypoints 90
copying debugging profiles
with the 3270 interface 45
with the web interface 27
copying pages
partitions 12
create Java debugging profile screen
application debugging profile manager
3270 interface 51
Create Java profile page
application debugging profile manager
web interface 32
create LE debugging profile screen
application debugging profile manager
3270 interface 47
Create LE profile page
application debugging profile manager
web interface 29
creating debugging profiles
with the 3270 interface 42
with the web interface 24
CSA (common system area)
warm start 62
CSD (CICS system definition) file
cold start 61

## D

data set backup
BWO processing 135
data set name blocks (DSNBs)
recovery 111
data set restore
BWO processing 136
data sets
backup while open 124
eligible for BWO 125
DATFORM, system initialization parameter 12
debugging profile manager
3270 interface 39
activating debugging profiles 43
changing debugging profiles 44
copying debugging profiles 45
create Java debugging profile screen 51
create LE debugging profile screen 47
creating debugging profiles 42
deleting debugging profiles 45
inactivating debugging profiles 44
list debugging profiles screen 39
set LE debugging display device screen 57
setting the display device 56
debugging profile manager *(continued)*
3270 interface *(continued)*
view Java debugging profile screen 54
view LE debugging profile screen 54
viewing debugging profiles 44
web interface 19
activating debugging profiles 25
changing debugging profiles 27
configuring 19
copying debugging profiles 27
Create Java profile page 32
Create LE profile page 29
creating debugging profiles 24
deleting debugging profiles 28
inactivating debugging profiles 26
List profiles page 20
Set LE display device page 38
setting the display device 36
View Java profile page 36
View LE profile page 35
viewing debugging profiles 26
debugging profile manager, application 18
debugging profiles
activating
with the 3270 interface 43
with the web interface 25
changing
with the 3270 interface 44
with the web interface 27
copying
with the 3270 interface 45
with the web interface 27
creating
with the 3270 interface 42
with the web interface 24
deleting
with the 3270 interface 45
with the web interface 28
inactivating
with the 3270 interface 44
with the web interface 26
viewing
with the 3270 interface 44
with the web interface 26
deleting debugging profiles
with the 3270 interface 45
with the web interface 28
DFH$OFAR 161
DFHCICSP 8
DFHKE1799 message 68, 69, 71
DFHSI1572 103
disaster recovery
considerations 143
high availability 155
PPRC and XRC 155
RRDF 158
testing 144
tiered solutions 145
disaster recovery facilities 159
CICS emergency restart 160
MVS system logger support 159
OFFSITE, system initialization parameter 161
disaster recovery facilities *(continued)*
support for RLS-mode data sets 161
display device
setting
with the 3270 interface 56
with the web interface 36
DSNBs, data set name blocks
recovery 111
dynamic RLS restart 99
dynamic transaction backout
failure during immediate shutdown 69

## E

emergency restart
process 64
resynchronization of z/OS Communications Server messages 64
ENF, event notification facility
notifying CICS of SMSVSAM restart 99
error message in partition 12
EXEC CICS PERFORM SHUTDOWN command 66
EXEC CICS PERFORM SHUTDOWN IMMEDIATE command 66
extended storage cushion
warm start 62

## F

FCT (file control table)
warm start 62
file closing
BWO processing 133
file control
file states on warm start 62
warm start 62
file opening
BWO processing 131
File resources
Dump table 105
LIBRARY resources 105
Resource definitions 105
START=COLD 105
Temporary Storage 105
Terminal control resources 105
Transient data 105
forward recovery
BWO processing 137
recovery point for BWO 137
forward recovery logging
BWO processing 136
forward recovery logs 86

## G

global catalog
file control table (FCT) warm start 62
monitoring options 63
statistics options 63
warm keypointing during CICS system 68
warm start resource definition 61

global catalog *(continued)*
warm-start-possible indicator 68

# I

ICE (interval control element)
warm start 63
identifying terminals 10
immediate shutdown 91
inactivating debugging profiles
with the 3270 interface 44
with the web interface 26
initialization
cold start process
file control 107
monitoring and statistics 109
temporary storage 109
terminal control resources 108
options 104
warm start process
file control 110
LIBRARY resources 113
monitoring 114
programs, mapsets and partitionsets 113
statistics 114
temporary storage 111
transactions 113
transient data 112
URIMAP definitions 116
virtual hosts 116
intrapartition data set
warm start 62
invoking a transaction 6
IRC (interregion communication)
CICS system 68
IST967I 103

# J

JCL to submit CICS commands 3

# K

kernel domain
CICS system termination processing 68, 71
keypoints
warm 90
keyword, minimum abbreviation 8

# L

LIBRARY definitions
cold start 60
list debugging profiles screen
application debugging profile manager
3270 interface 39
List profiles page
application debugging profile manager
web interface 20
locks 78
LUs, logical units
console as a CICS terminal 14

# M

machine check 67
manager, application debugging profile 18
managing UOW state 82
mapset definitions
warm start 63
master terminal operator
duties 11
restriction 11
types of terminal 11
maximum task values
warm start 62
messages
chaining in partition 12
DFHKE1799 68, 69, 71
purging 12
replying to messages 4
replying to messages from transactions 5
suppressing 3
suppressing and rerouting 6
terminating in partition 12
z/OS Communications Server 64
MNPS 100, 101, 102
MODEL definitions
cold start 60
MODIFY command 2, 15
monitoring
warm start 63
multinode persistent sessions 100, 101, 102
multiple console support 14
MVS automatic restart manager 118

# N

NetView
DFH$OFAR, 161
NOPS 100, 103
normal shutdown 88
notation, syntax 7

# O

off-site recovery
support for RLS-mode data sets 161
OFFSITE, system initialization parameter 161
operating system console as a terminal 14
operating system failure 67
operating system requested shutdown 92
operations, automated 2
operator security 9

# P

PA1 print key 12
page copying 12
page retrieval
and partitions 12
page retrieval function
new PROFILE definition, DFHCICSP 8
partitions, BMS 12
PERFORM SHUTDOWN command 88
PERFORM SHUTDOWN IMMEDIATE command 91
persistent sessions 100, 101, 102, 103
persistent sessions delay interval 101, 102, 103
personnel for disaster recovery 159
PLT (program list table)
first stage shutdown programs 67
second stage shutdown programs 67
power failure 67
PPRC 155
print (PA1) key 12
PROFILE definitions
cold start 60
warm start 62
profile manager, application debugging 18
profile manager, debugging
web interface 19
configuring 19
program check
cause of immediate shutdown 66
PROGRAM definitions
cold start 60
warm start 63
program library
cold start resource definition 61
warm start resource definition 61
programs
PLT first stage shutdown 67
PLT second stage shutdown 67
PSDINT 101, 102, 103
PSTYPE 100, 101, 102, 103

# Q

quiesce stages of normal shutdown 88

# R

recording of recovery information
forward recovery logs 86
recording of replication information
replication logs 87
recovery
bundle resources 116
recovery manager 81
coordinating recoverable remote conversations 84
coordinating resource updates 83
managing the UOW state 82
shunted state 77
recovery point for BWO 137
region exit time interval value
warm start 62
Remote Recovery Data Facility 153
REPLY command 15
REPLY command, responding to console messages 5

resend slot
re-presentation of z/OS Communications Server messages 64
resource definition
cold start 61
warm start 61
resource definitions
profile, DFHCICSP 8
restart
BWO processing 134
retrieve a page 12
RRDF 153
runaway time interval value
warm start 62

## S

security considerations
for BWO 130
security, operator 9
set LE debugging display device screen
application debugging profile manager
3270 interface 57
Set LE display device page
application debugging profile manager
web interface 38
SET VTAM 102
setting the display device
with the 3270 interface 56
with the web interface 36
shunted state, of unit of work 77
shunted unit of work 81
shutdown
BWO processing 134
immediate 91
normal 88
requested by operating system 92
uncontrolled 92
single-node persistent sessions 100, 101, 102
SNA ACB at startup 65
SNPS 100, 101, 102
START=COLD specification 104, 105
starting a transaction 6
starting CICS regions 59
statistics
termination 69
warm start 63
storage cushion
warm start 62
STORECLOCK value warm start 63
subsystem interface
termination 67, 69
supervisory terminal operator
duties 10
identifying terminals 10
SUPRID option 10
syncpoint
general description 79
syntax notation 7
system activity keypoints
description 86
system initialization parameters
OFFSITE 161
system log
for backout 85
information recorded on 85
system startup 59
system warm keypoints 90
systems administration
for BWO 129

## T

TCT (terminal control table)
warm start 63
temporary storage 63
warm start 63
terminal control
CICS system processing 68
termination processing 69
TERMINAL definitions
cold start 60
terminal I/O errors, recovery
terminal error program immediate shutdown 91
terminal list table (TLT) 10
terminal operator
duties 10
master 11
supervisory 10
transactions 9
terminals
identifying 10
paging 12
terminate a logical message 12
termination task
priority change to zero 67
processing 67
the z/OS Communications Server
persistent sessions support 100
tie-up record 136
time sharing option (TSO)
using command lists 2
TLT (terminal list table) 10
TPEND 102
transaction abend
during immediate shutdown 69
TRANSACTION definitions
cold start 60
warm start 62
transaction list table (XLT) 67, 88
transactions
canceling 7
identification codes 10
initiating from console 15
invocation of 6
security key 10
transactions allowed during normal shutdown 88
transactions with operator interface
CADP 39
transient data
intrapartition warm start 62
TSO (time sharing option)
using command lists 2
TSO console as a CICS terminal 17
type-of-restart indicator
emergency-restart-needed 91
operation of 97
warm-start-possible 89
TYPETERM definitions
cold start 60

## U

uncontrolled shutdown 92
unit of recovery (see unit of work) 77
unit of recovery descriptor (URD) 63
unit of work
atomic 84
managing state of 82
overview 77
shunt 81
shunted state 77
unshunt 81
updates to local resources 83
uppercase input to transactions 8
URD (unit of recovery descriptor) 63
URIMAP definition 116
user journals 87

## V

view Java debugging profile screen
application debugging profile manager
3270 interface 54
View Java profile page
application debugging profile manager
web interface 36
view LE debugging profile screen
application debugging profile manager
3270 interface 54
View LE profile page
application debugging profile manager
web interface 35
viewing debugging profiles
with the 3270 interface 44
with the web interface 26
virtual host 116
VSAM CI (or CA) split
effect on BWO 126
VSAM upgrade set
effect on BWO 133

## W

warm keypoints
warm start resource definition 61
warm restart
rebuilding the CICS state 110
requirements for restart 110
START=AUTO 110
warm start
autoinstalled terminals 63
basic mapping support (BMS) 63
common system area (CSA) 62
file control table (FCT) 62
file states 62
installed program definitions 63
interval control elements 63
intrapartition transient data 62
logical end of day 63

warm start *(continued)*
mapset definitions 63
monitoring 63
partial 61
process 61
profile definitions 62
resource definition 61
statistics collecting interval 63
statistics collecting status 63
STORECLOCK value 63
temporary storage 63
terminal control table (TCT) 63
transaction definitions 62
unit of recovery descriptor (URD) 63
warm-start-possible indicator 68
web interface
application debugging profile manager 19
workload policies 121

# X

XLT (transaction list table) 67, 88
XMEOUT, global exit for message handling 6
XRC 155
XRF 102

# Z

z/OS Communications Server
persistent sessions support 101, 102, 103
z/OS Communications Server messages
resynchronization after emergency restart 64
z/OS Communications Serverthe Communications Server
persistent sessions support 102

