CICS Transaction Server for z/OS Basic Tailoring

(Course Code CI20)

Student Notebook

ERC 7.0

IBM Certified Course Material

Trademarks

IBM® is a registered trademark of International Business Machines Corporation.

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

AD/Cycle®	AIX®	C/370™
CICS®	CICS/ESA®	CICS/MVS®
CICS/VSE®	CICSPlex®	COBOL/370™
CUA®	DB2®	DFSMSdss™
DFSMShsm™	Enterprise Systems Architecture/390®	ESCON®
IMS™	IMS/ESA®	InfoWindow®
iSeries™	Language Environment®	Lotus®
MQSeries®	MVS™	MVS/DFP™
MVS/ESA™	NetView®	OS/2®
OS/390®	Parallel Sysplex®	PR/SM™
RACF®	RETAIN®	RMF™
S/390®	SAA®	SP™
SQL/DS™	Systems Application Architecture®	System/390®
Tivoli®	VisualAge®	VM/ESA®
VTAM®	WebSphere®	z/OS®
zSeries®	1-2-3®	3090™

PS/2® is a registered trademark of Lenovo in the United States, other countries, or both.

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

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

Intel and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

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

August 2005 Edition

Contents

Trademarks

The reader should recognize that the following terms, which appear in the content of this training document, are official trademarks of IBM or other companies:

IBM® is a registered trademark of International Business Machines Corporation.

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

AD/Cycle®	AIX®	C/370™
CICS®	CICS/ESA®	CICS/MVS®
CICS/VSE®	CICSPlex®	COBOL/370™
CUA®	DB2®	DFSMSdss™
DFSMShsm™	Enterprise Systems Architecture/390®	ESCON®
IMS™	IMS/ESA®	InfoWindow®
iSeries™	Language Environment®	Lotus®
MQSeries®	MVS™	MVS/DFP™
MVS/ESA™	NetView®	OS/2®
OS/390®	Parallel Sysplex®	PR/SM™
RACF®	RETAIN®	RMF™
S/390®	SAA®	SP™
SQL/DS™	Systems Application Architecture®	System/390®
Tivoli®	VisualAge®	VM/ESA®
VTAM®	WebSphere®	z/OS®
zSeries®	1-2-3®	3090™

PS/2® is a registered trademark of Lenovo in the United States, other countries, or both.

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

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

Intel and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

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

Course Description

CICS Transaction Server for z/OS Basic Tailoring

Duration: 4.5 days

Purpose

This course is designed to teach students how to install and tailor a CICS Transaction Server for z/OS system in order to be able to set up and start a new CICS region, and to act on basic requirements so as to support network devices to be connected to CICS and applications to be run under CICS control.

Moreover, the course provides thorough introductions to the concepts of security and recovery issues, which will enable students to participate in the appropriate administration activities to support new or existing installations of CICS TS.

Audience

System programmers installing and tailoring CICS Transaction Server for the first time.

Prerequisites

- A working knowledge of MVS (z/OS) and JCL.
- An introductory knowledge of CICS, which can be obtained from attending the *CICS Fundamentals* or *CICS Facilities and Education Planning* courses, or by reading the manual: *CICS Family: General Information, GC33-0155*.
- A basic knowledge of communications systems concepts and Systems Network Architecture concepts.
- A basic knowledge of VSAM concepts and Access Method services.
- Familiarity with handling TSO/ISPF and handling job output listings, preferably using SDSF.

Objectives

After completing this course, a student should be able to:

- Install CICS Transaction Server V3 (if familiar using SMP/E) and set up specific CICS regions ready to be started:
  - Identify sources of information necessary to do preinstallation planning
  - List the CICS TS data set requirements
  - Identify operating system requirements necessary to install CICS TS
  - Identify the pregenerated elements of CICS TS
- Using a given application description, define resources and customize the following:
  - Transactions and associated programs/map sets
  - VSAM files
  - 3270 terminals and autoinstall models supporting them
  - Transient data destinations
  - Shared temporary storage pools
  - CICS TS log streams
- Define the major functions and parameters associated with the following:
  - Initiation/termination
  - Recovery/restart; use of MVS log streams
  - Security
- Identify procedures required to install the first application
- Identify the functions of the basic system control parameters
- Identify the major elements of the statistics
- Identify requirements to collect monitor data
- List the functions of intercommunication facilities
- Identify sources of information necessary for basic problem determination
- Describe how the other MVS resource managers offered by IBM interface with CICS Transaction Server
- Describe the choices for invoking CICS programs and accessing CICS data from outside CICS

Contents

See objectives

Curriculum relationship

- CI01, or the appropriate distributed learning course, CS01, is a prerequisite to CI20.
- CI20 provides the base for all CICS TS system courses, the most important of which are:
  - CI21, *CICS TS Java Enablement*
  - CI29, *CICS TS Problem Determination*
  - CI72 and CI73, *CICS Web Enablement*
  - CI75 and CI76, *CICSPlex SM Introduction and Administration*
  - and all the courses covering specific CICS TS system subjects, such as Recovery (CI28), Security (ES84), DB2 Attach (CI24).

Agenda (Proposed)

Day 1

Welcome, including Unit 1 - Course Introduction
Unit 2 - CICS TS for z/OS Overview (including break)
Unit 3 - Installation and Verification
Lunch break
Unit 3 continued
Lab Exercise 1 - Using DFHISTAR
Coffee break
Unit 4 - Resource Definition Online
Lab Exercise 2 - Using the RDO Batch Utility

Day 2

Unit 5 - Supporting Terminal Access (including break)
Lab Exercise 3 - Defining Terminals
Unit 6 - Defining Transactions, Programs, and Map Sets
Lunch break
Lab Exercise 4 - Defining Transactions, Programs, and Map Sets
Coffee break
Unit 8 - Queuing Facility (TD and TS)

Day 3

Lab Exercise 5 - Define Files
Lab Exercise 6 - Define Queues
Unit 9 - CICS Intercommunication Services
Lunch break
Unit 9 - continued
Break
Unit 10 - System Control
Lab Exercise 9 - Using the Console Transaction CEKL

Day 4

Unit 11 - Storage Management and Storage Protection
Optional unit (Unit 18) - Accessing External Resource Managers' Data
Unit 12 - Startup and Recovery (Topic 1)
Lunch break
Unit 12 continued (Topics 2-4)
Lab Exercise 7 (optional) - Connect CICS Regions
Break
Lab Exercise 8 - Define Journal Models and Recoverable Files
Optional unit (Unit 17) - Sysplex-Related Functions

Day 5

Unit 13 - Security using an External Security Manager
Unit 14 - Basic Problem Determination
Unit 15 - Measurement and Evaluation

Unit 1. Course Introduction

What This Unit Is About

This introductory unit tells you how the course relates to the rest of the CICS curriculum.

What You Should Be Able to Do

After completing this unit, you should be able to:

- Name the units covered and the class schedule
- Explain where CICS TS for z/OS Basic Tailoring fits into the CICS curriculum.

How You Will Check Your Progress

Not applicable to this introductory unit.

References

- Refer to the course offering as announced by IBM in your country.

1.1 Course Introduction

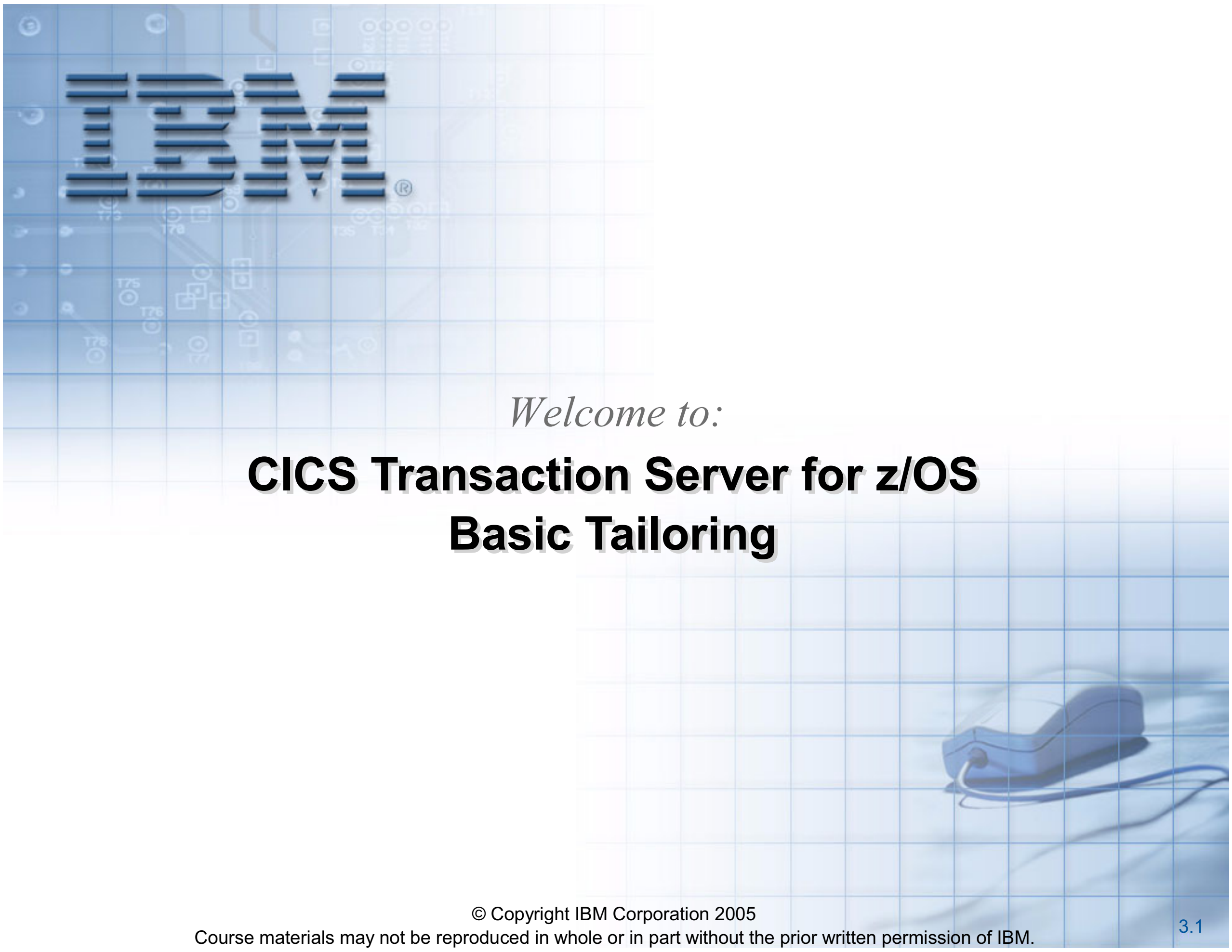


Figure 1-1. CICS Transaction Server for z/OS Basic Tailoring CI207.0

Notes:

The CI20 *CICS Transaction Server for z/OS Basic Tailoring* course teaches how to set up one or more CICS TS regions in order to support applications using structured languages.

By way of lectures and lab exercises, it provides an understanding of what system functions are involved in executing online applications, and how they may be tailored to meet the needs of users and applications.

Course Prerequisites

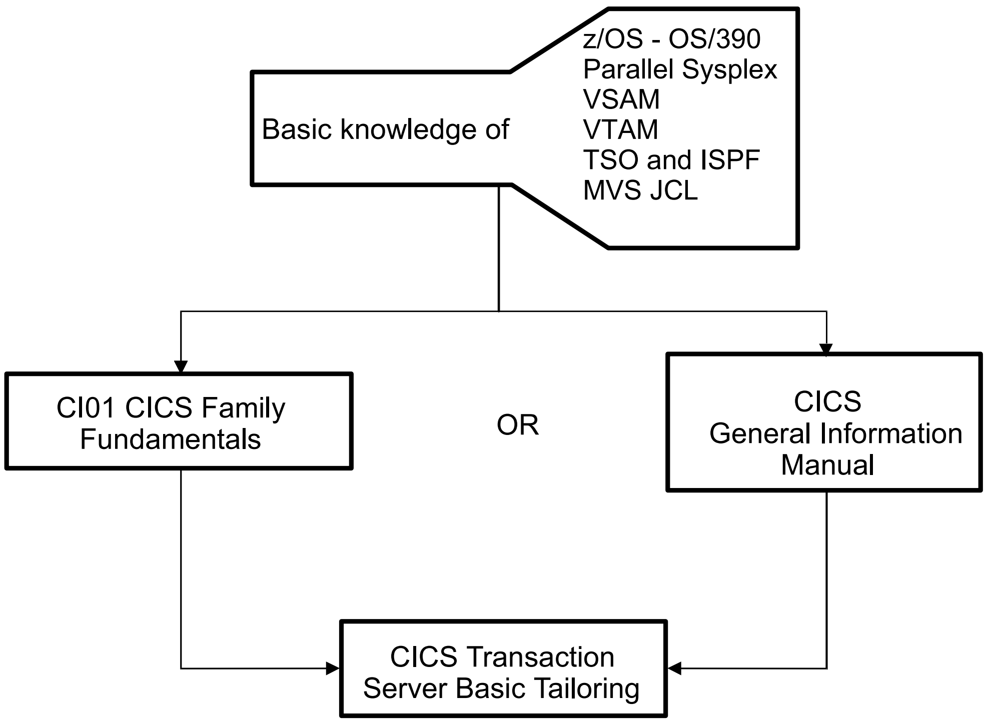


Figure 1-2. Course Prerequisites CI207.0

Notes:

- Only a very brief overview of CICS fundamentals will be provided, so basic knowledge is assumed, as provided by the *CICS Fundamentals* class or the brochure *CICS Family: General Information, GC33-0155*.
- Some batch lab exercises will be run in TSO and ISPF.
  JCL will be supplied for all the lab batch runs, but some general MVS knowledge is helpful.
- Basic knowledge of VSAM and CICS is required, as is a general understanding and knowledge of the Parallel Sysplex concepts.

References:

GC33-0155 *The CICS Family: General Information*
GC28-1208 *Sysplex Overview: An Introduction to Data Sharing and Parallelism*

Curriculum Relationship

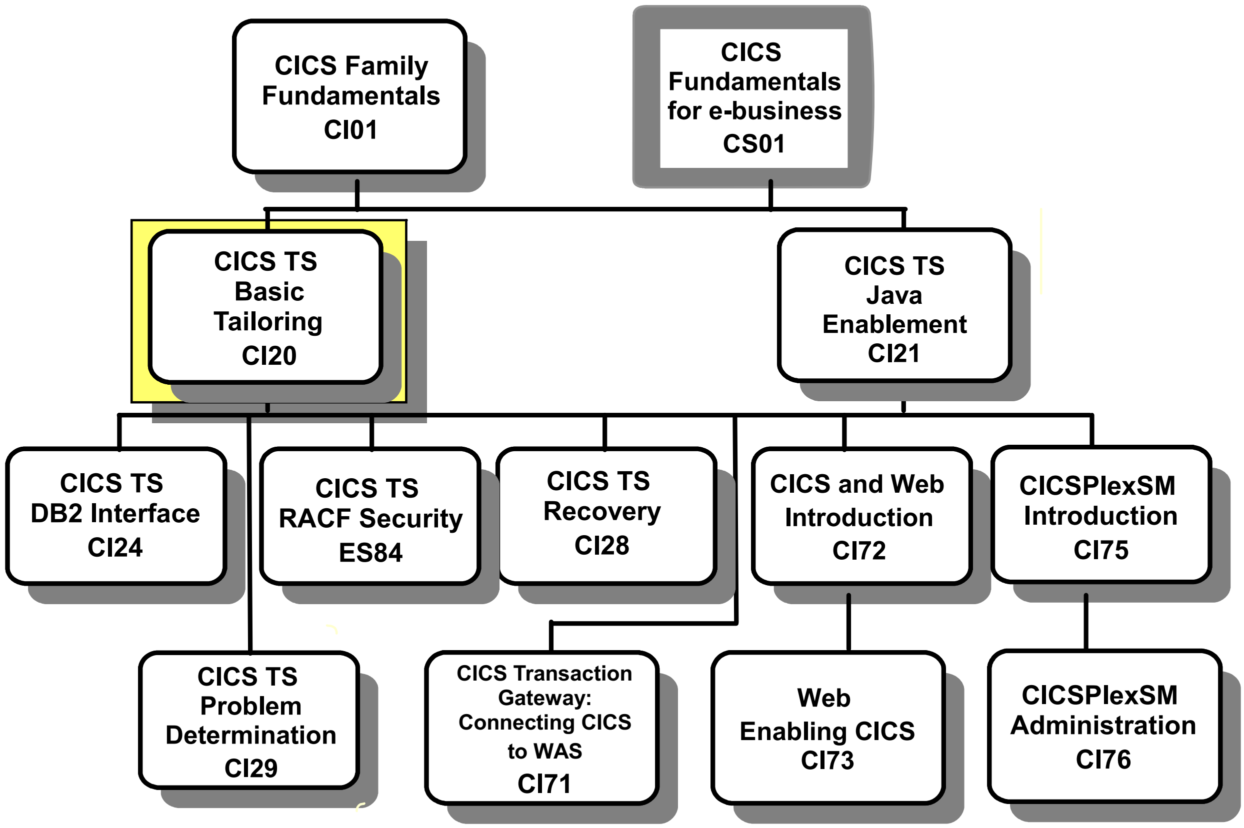


Figure 1-3. Curriculum Relationship CI207.0

Notes:

The visual shows how and where our course fits into the IBM IT Education Services worldwide curriculum of CICS TS system programming.

- The graphic shows the international course numbers, which may be *slightly* modified for your country's offerings.
- Not all courses are available in all countries.

Schedule for Week

Monday	Tuesday	Wednesday	Thursday	Friday
Introduction CICS TS for z/OS Overview CICS TS for z/OS Installation Region Setup and Verification	Supporting Terminal Access Lab Exercise 3: Administering Terminals Defining Transactions, Programs, and Map Sets	Lab Exercise 5: Defining Files and (optionally) LSR Pools Lab Exercise 6: Defining Queues CICS Intercommunication Services	Storage Management and Storage Protection Optional Unit: Accessing External Resource Managers' Data System Startup and Termination Restart and Recovery Issues	Security using an External Security Manager Basic Problem Determination Measurement and Evaluation Optional Unit: Related CICS Product Programs
Lab Exercise 1: Setting Up your CICS Region Resource Definition Concepts Lab Exercise 2: RDO Batch Utility - Startup CICS	Lab Exercise 4: Define Transactions and Programs File and LSR Pool Definitions Queuing Facilities (TD and TS)	Lab Exercise 8: Activating a Web Services Application System Control Lab Exercise 9: Using CEKL Demonstration	Optional Lab Exercise 7: Connecting Systems Lab Exercise 10: Defining Recoverable Files and Journal Models Optional Unit: Sysplex-Related Functions	

Figure 1-4. Schedule for Week CI207.0

Notes:

- This is a suggestion, as made by the course developer.
- While the first units will most certainly need to be covered in the sequence in which they appear in this material, some flexibility may be used for covering later units, and particularly for the optional topics.
- One basic principle is that balancing lectures and practical exercises will be attempted throughout the course week.

Unit 2. CICS Transaction Server for z/OS Overview

What This Unit Is About

This unit provides a short technical introduction to the CICS Transaction Server family with its role in both traditional (host-oriented) and modern (service-oriented architecture SOA) IT infrastructures.

The second topic shows you how the CICS Transaction Server for z/OS region is organized, and how a task flows through CICS TS from terminal entry to message reply.

A third topic provides an overview of the products that make up the CICS Transaction Server for z/OS.

What You Should Be Able to Do

After completing this unit, you should be able to:

- Identify the role of CICS TS in different approaches of Information Technology (IT) infrastructures
- Identify the major CICS TS for z/OS functional components
- Describe the major areas of a CICS TS address space and the parameters that influence the sizes of these areas
- Describe, at a high level, the flow of a typical CICS task
- Identify the most important system resources used by a task.

References

CICS Family: General Information, GC33-0155

Why CICS? (Brochure), G325-5439

Unit Objectives

After completing this unit, you should be able to:

- Identify the role of CICS TS in different approaches of Information Technology (IT) infrastructures
- Identify the major CICS TS for z/OS functional components
- Describe the major areas of a CICS TS address space and the parameters that influence the sizes of these areas
- Describe, at a high level, the flow of a typical CICS task
- Identify the most important system resources used by a task

Figure 2-1. Unit Objectives CI207.0

Notes:

2.1 What is CICS?

CICS as a Transaction Management System

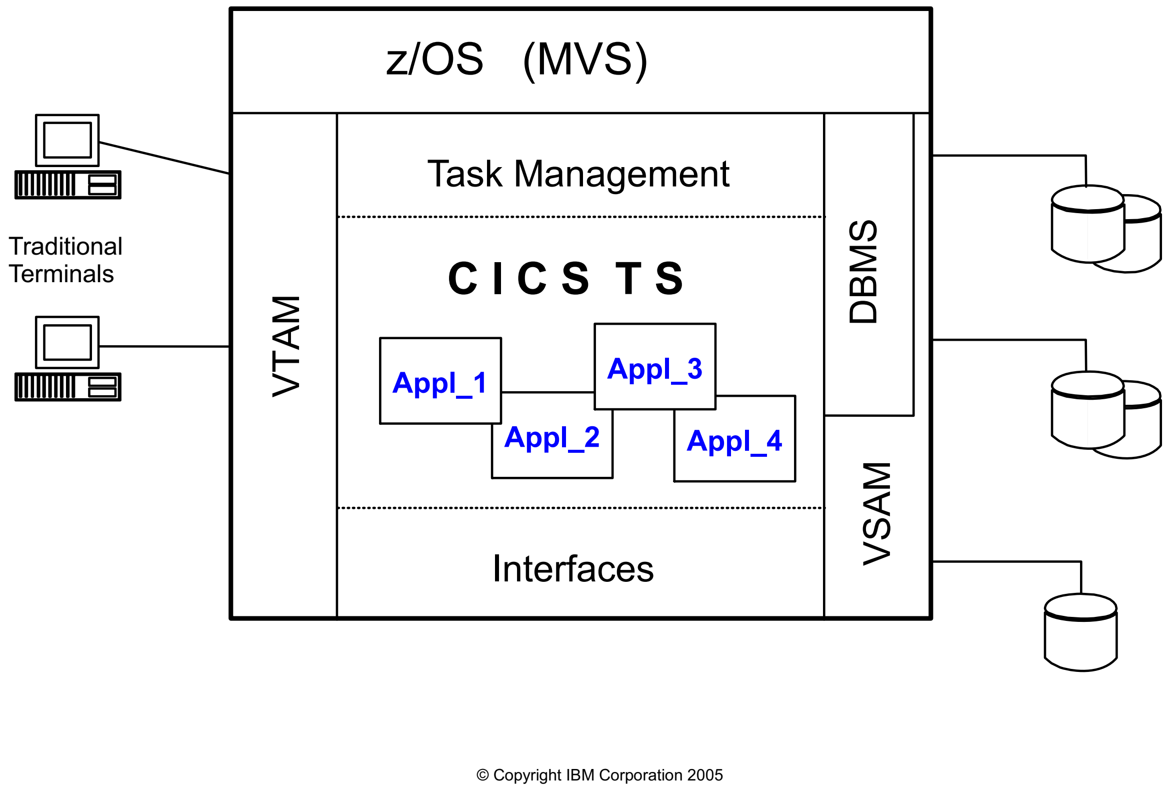


Figure 2-2. CICS as a Transaction Management System CI207.0

Notes:

This visual covers the more traditional role of CICS in a host-oriented IT infrastructure, where users are working mainly with “dumb” terminal devices that are directly connected to the host, and the whole application, including presentation logic, business logic, and data access is under control of CICS.

- CICS is a general-purpose Data Base/Data Communications (DB/DC) system. The term DB/DC describes the type of processing done by an online system.
- Generally, online systems process transactions which include:
  - Receiving a message from a terminal
  - Accessing data from a file or database
  - Sending a reply back to the terminal
- Users log on to a CICS region and are — more or less — aware of CICS hosting their applications.
- CICS uses the services of other products, such as VSAM and VTAM, and provides interfaces to DB2, IMS DL/I, and RACF.

"What is CICS?" - Possible Answers:

An Online Transaction Manager

Terminal users access and change online data, in other words, data that is available concurrently to lots of users with a high degree of actuality, through single processes called transactions.

An Application Server

Applications may issue requests for resources, services, or both, in order to fulfil the users' demands. These requests are served by CICS, and are transparent to the applications.

Multi-Tasking System

Multiple transactions may be operational at one time, accessing different or the same resources. CICS controls their concurrent processing and takes cares of the integrity between them.

A Resource Manager

Applications do access and use resources, but do not have to manage and control them: CICS "owns" terminals, files, virtual storage, and so on, that it handles on behalf of the application.

A z/OS Subsystem

It uses the services and functions of other products such as VTAM and VSAM (part of DFSMS), and provides interfaces to Data Base Management Systems (DBMS), DB2, and DBCTL (IMS), as well as to the security product RACF.

A Client/Server Application Enabler

In addition to the services mentioned above, CICS enables applications to distribute their functions easily between various operating system environments.

CICS as an Application Server

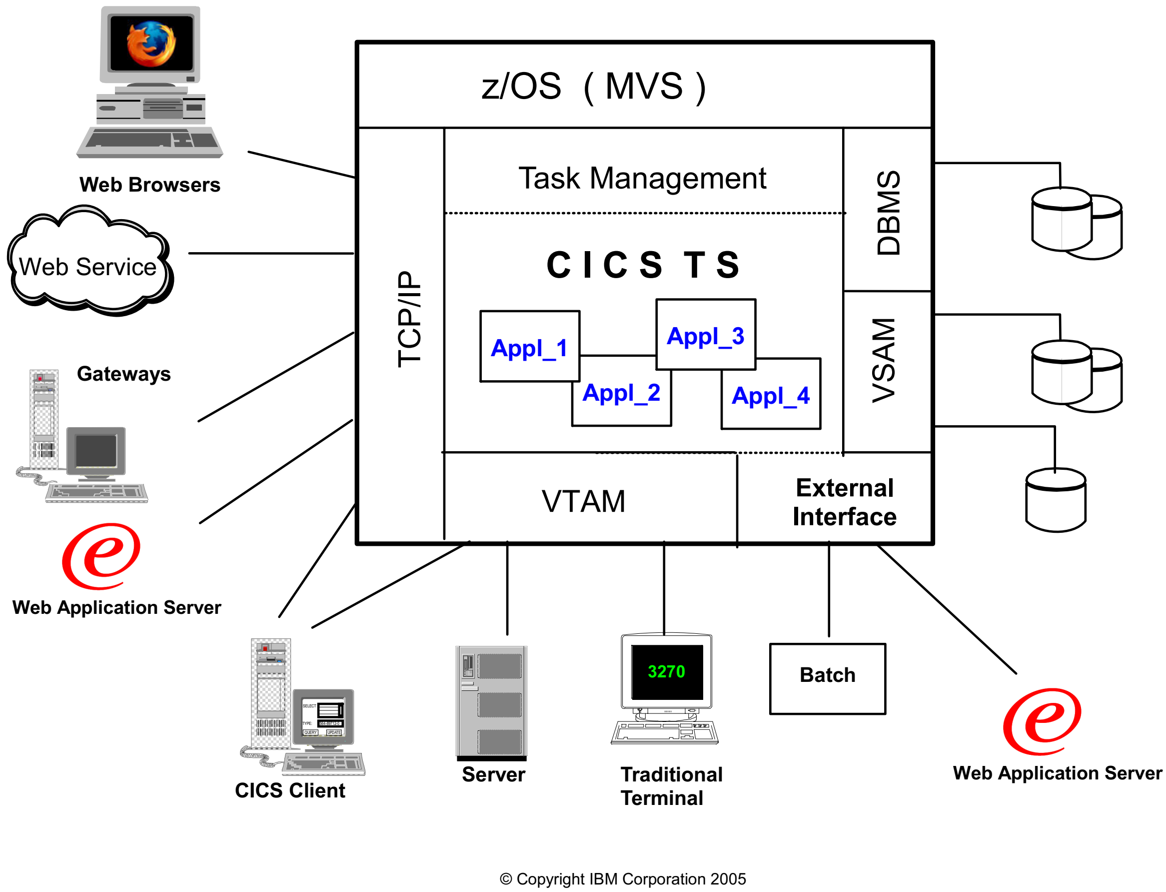


Figure 2-3. CICS as an Application Server CI207.0

Notes:

As an **application server**, CICS provides access to environment data for products on different platforms under the control of transaction management, ensuring integrity and consistency to the data accessed.

- Users may work with workstations or web browers that provide (graphical) user interfaces. Certain parts of the applications' logical functions may run on these workstations or on a web application server. Only portions of an application may run under CICS control, primarily those that deal with the manipulation of critical data. Usually, users are not aware of having anything to do with CICS; normally they won't even know that there is such a thing!
- From CICS's point of view, most requests for service that result in launching CICS transactions may not be issued by users entering transaction codes, but through various mechanisms by systems (applications) that are connected to CICS via the network or through interfaces supported by z/OS.
- Note that the latest releases of CICS TS mainly concentrated on providing application server support for Java-based applications. Setting up and administering these services is covered by course CI21.

The CICS Application Programming Interface

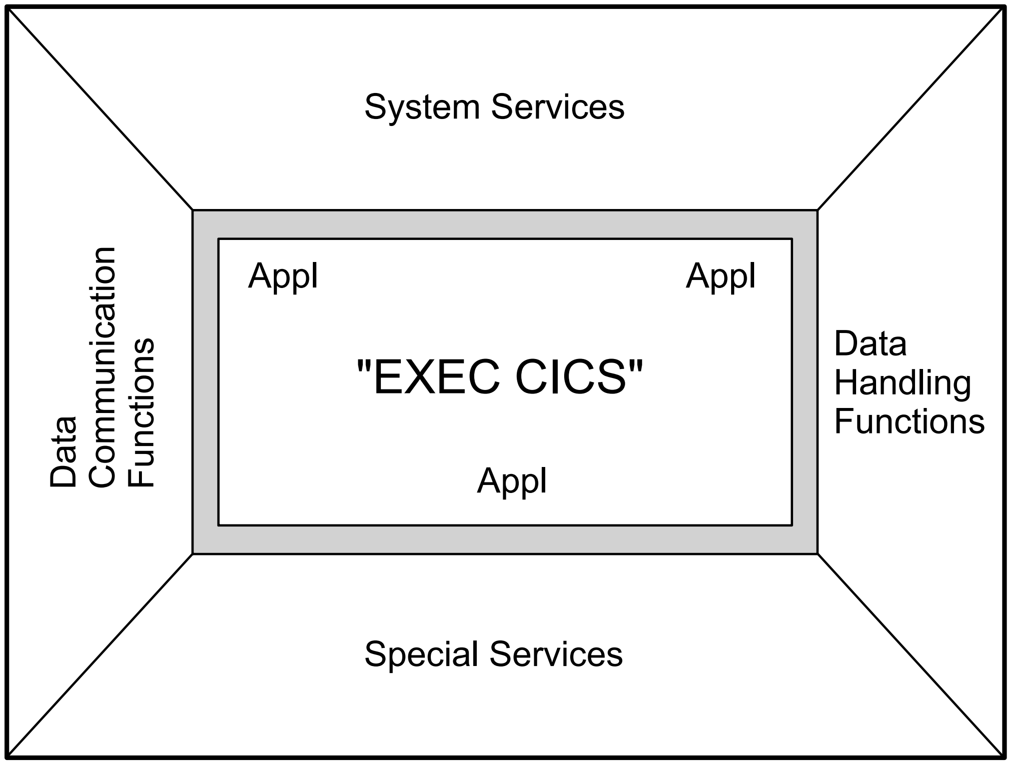


Figure 2-4. The CICS Application Programming Interface CI207.0

Notes:

- Traditional CICS application programs are “embedded” in CICS, in the way that they direct their requests for resources and services to CICS, particularly those requests that finally address some “external” instance, such as the operating system or networking facilities.
  CICS then forwards these requests to the appropriate external instance, which in most cases is not aware of the CICS internal tasks, but just “knows” CICS as its client.
- Application requests for service look like this:
  ```
  EXEC CICS  function  parameter(s)
  ```
- The common **EXEC CICS API** is the main characteristic of all members of the **CICS family**.

Note that the first bullet statement from above is not true for CICS Java programs and Enterprise Java Beans. Starting with CICS TS V2, a concept was introduced that also allows non-Java CICS applications to directly access external services. Application programs that make use of this option are required to be *threadsafe*. Refer to the *CICS Application Programming Guide, SC34-6433* for details.

2.2 CICS Task Flow and Functional Components

CICS Task Flow

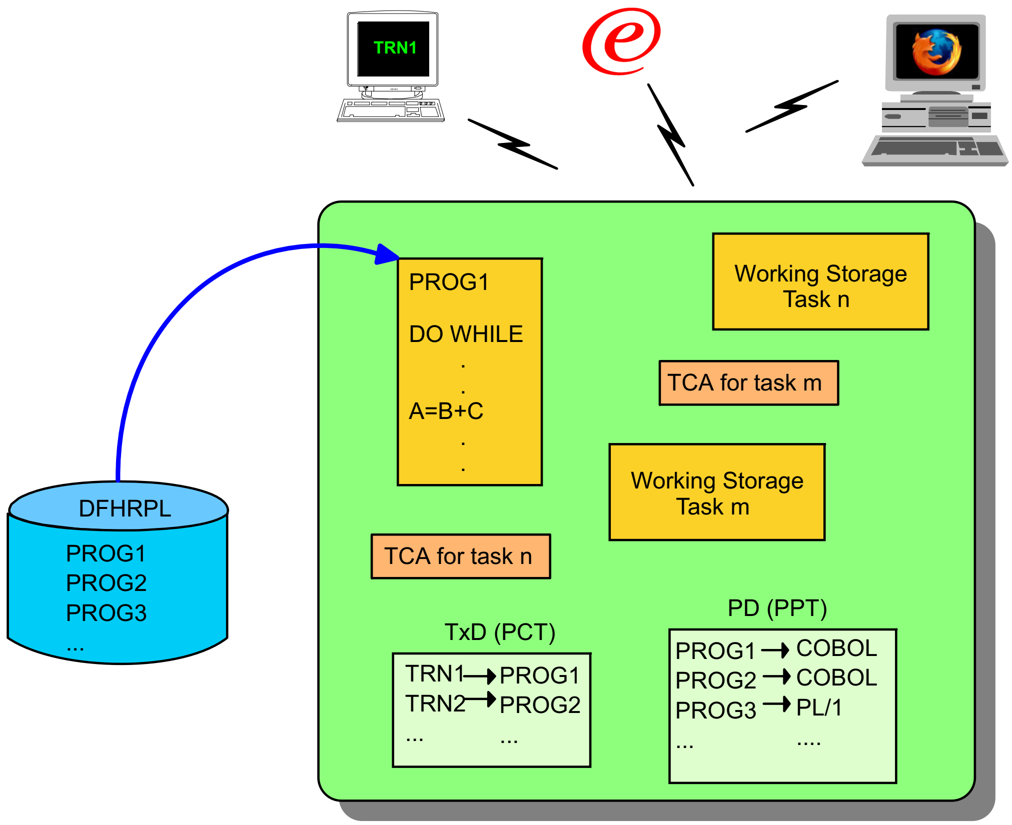


Figure 2-5. CICS Task Flow CI207.0

Notes:

Transaction User-initiated request for work, either entered from a terminal, or sent from a connected device or system as a request for service.

Task Internal unit of work initiated by a transaction.

CICS Task Flow

- Transaction TRN1 is invoked via the network, either entered by a terminal user or a browser or a web application server or issued by a connected system or application. VTAM or TCP/IP receives the message and passes it to one of the CICS protocol handler. The protocol handler copies the message into a buffer for later processing. Transaction Manager (XM) validates the transaction ID using the Transaction Directory (TXD). CICS issues an error message for invalid transactions.
- If TRN1 is secured, CICS calls RACF or another External Security Manager (ESM) to perform authentication and set up security privileges. From a 3270 terminal, the user must first sign on using a CICS transaction. Security credentials may be transmitted to

CICS from a distributed system, along with the transaction request. On TCP/IP networks digital certificates can be used.

- The Transaction Directory (TXD) defines a valid transaction in the CICS system. Since transaction TRN1 is valid, XM creates a Task Control Area (TCA). Storage Management (SM) acquires the areas from common storage within CICS address space named Dynamic Storage Area(s) (DSA).
- Next, your task needs an application program. The TXD also includes the name of the first program to be executed, in this case PROG1. The Program Directory (PD) defines each CICS and user application program (other than nucleus management modules). If the program has not already been loaded, the Program Domain loads it into the DSA from one of CICS's Relocatable Program Load Libraries (DFHRPL).
Storage Management acquires space for the program from the DSA.
- The Program Domain then copies PROG1's working storage into an independent area of storage. This allows PROG1 to be shared by more than one task. Having a unique copy of working storage for each task makes command-level CICS programs reentrant or pseudo-reentrant, allowing a single copy to be shared by multiple tasks.
- The EXEC Interface Program (EIP) intercepts application program requests and returns exception conditions and other information in the Execute Interface Block (EIB).
- When PROG1 issues an EXEC CICS READ to a file, the File Control Program (FCP) issues a request to VSAM based upon the data set information in the FCT. VSAM reads the record. FCP passes it to the program in the requested area.
- Changes to the VSAM data set may be logged for recovery purposes.
- In today's systems, the account information is likely to be kept within a DB2 table, in such a case the user application would use the standard SQL interface to access the account data. CICS would then communicate with DB2 through ERM support.
- Traditional applications that are fully controlled by CICS often use Basic Mapping Support to send formatted data to the terminal, or in other cases, tooling or CICS-generated code for HTTP, XML, or SOAP.
- If a program sends some data to a user interface and then issues a (EXEC CICS) RECEIVE command, all task-related storage is held in the DSA. This is called *conversational programming*. If, on the other hand, the task sends data to the user with no subsequent receive, it is a *pseudo conversational* task. CICS frees the storage areas, and the programmer passes any data required by the next task in a CICS Communication Area (COMMAREA) or containers and channels (CICS TS 3.1 and later).
- When the task ends, control is returned to CICS. Task-related storage in the DSA is released.
- Actually, many tasks may be active at a time. Whenever one task requests an input or output operation, CICS issues a task WAIT and allows another user's task to run.

CICS Management Functions

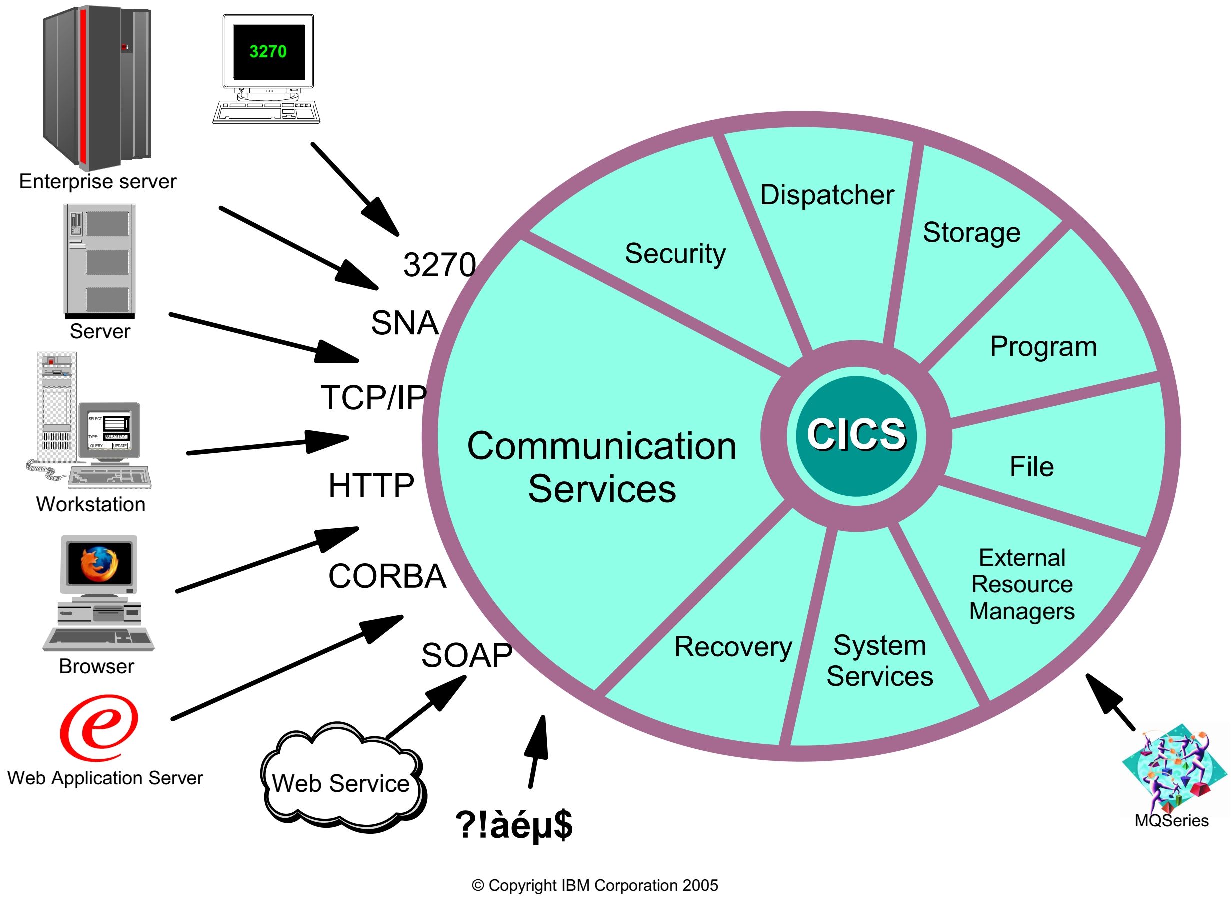


Figure 2-6. CICS Management Functions CI207.0

Notes:

CICS Management Functions

- **Communication Services.** Most traditional CICS applications start when an operator enters data from a terminal. Terminal management allows your applications to send or receive messages.
  We already stated that, by acting as a back-end server for applications hosted by various environments, transactions may enter the CICS region as service requests through the network or through a z/OS-based facility.
- **Security Management.** Access to transactions and other CICS resources can be restricted by permission profiles defined to an external security management system (for example, RACF). For a production system, this is the regular mode of operation. A terminal user has to sign on before the system will run transactions on their behalf. For alternative ways of entering transactions, the determining of a userid that is to be associated with the transaction request depends on the facilities and functions used.
- **Task Management.** CICS provides its own multi-user or multi-thread environment. Additional z/OS tasks are created for special purposes.

- **Program Management.** A typical CICS system supports many different user application programs. Loading and accessing these programs is controlled by this management function.
- **File Management.** Some applications use files to store data. File management provides access to VSAM and direct access (BDAM) data sets.
- **External Resource Managers.** Some applications use databases. External Resource managers provides access to IBM databases (DB2, DL/I) or non-IBM databases (ORACLE, SYSBASE...). External Resource Managers also provide access to WebSphere MQ.
- **Queue Management.** Some applications have the requirement to store data in a queue for later retrieval. CICS supports two methods of queuing.
- **Recovery Management.** You may ask CICS to protect selected resources, in the event of unexpected termination of the transaction or the system.
- **System Services**. This group of system programs provides such generalized services as:
  - **Storage Management** - acquiring storage in the DSA as CICS tasks need it.
  - **Time Management** - a service that allows CICS tasks to request that certain actions be performed at a certain time of day, or after an elapsed period of time.
  - **Trace Management** - a service that records calls to CICS management functions, which can be very helpful in problem determination.
  - **Application Program Interface (API)** - a “layer” between application programs and CICS management functions.

CICS Domain Structure

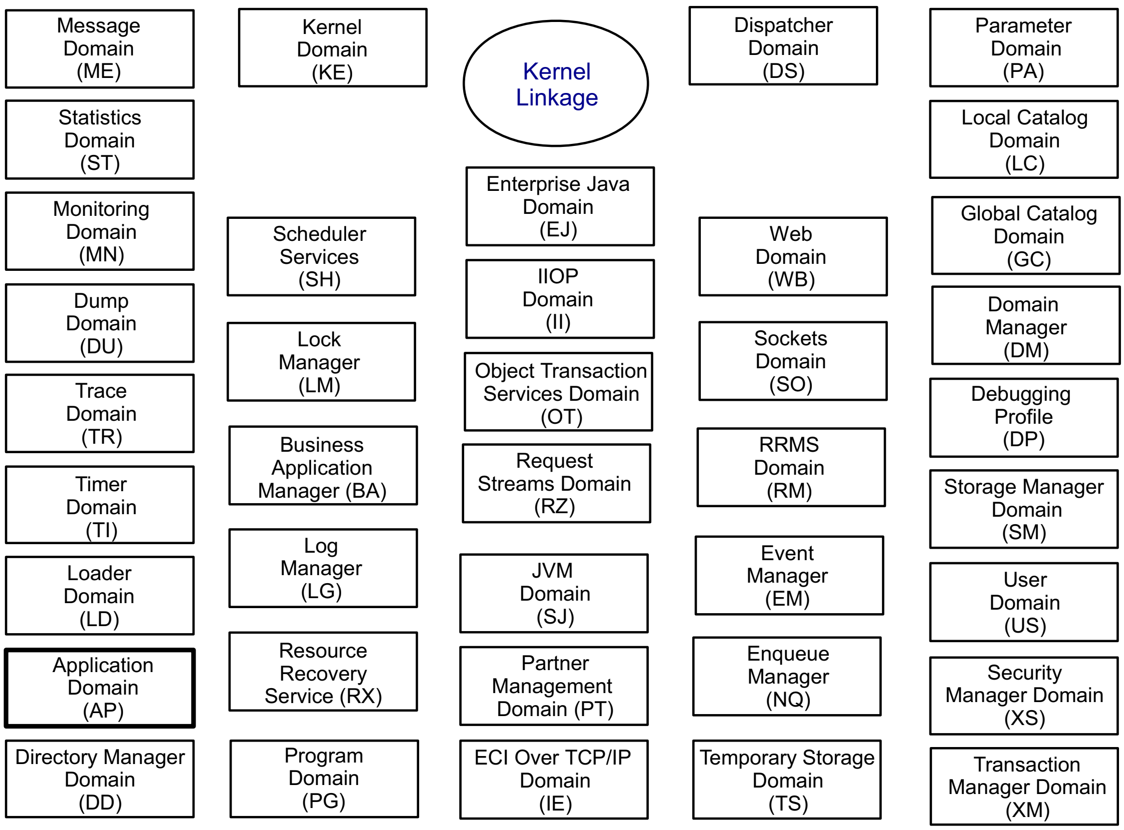


Figure 2-7. CICS Domain Structure CI207.0

Notes:

Internal CICS code has been structured into **domains**; with each release, some new domains have been added.

- CICS requests services by a *domain call* rather than direct branching. This improves reliability and maintainability.
- The Application Domain (AP) has a special status among the circle of domains:
  - A number of functions requested directly by application programs run in the AP, such as EXEC interface processing, and also terminal-oriented, file control, and transient data queuing services.
  - Parts of the AP are not yet restructured, but it interfaces with other domains according to domain architecture rules.
- The domain architecture is transparent externally, except that many CICS messages identify which domain produced the message.
- *CICS Diagnosis Reference, LY33-6110*, contains detailed documentation of all domains and all of their functions and interfaces.

CICS Region and the DSA

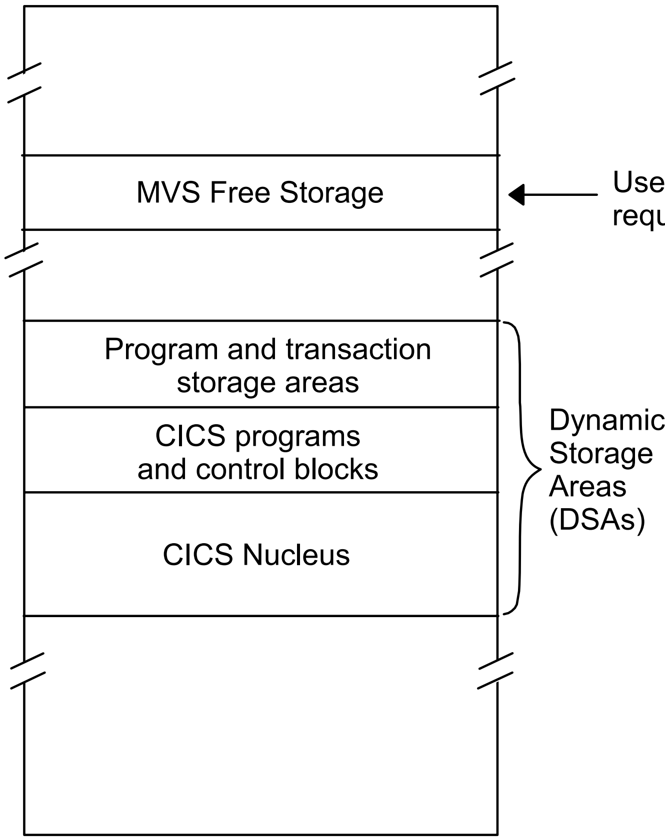


Figure 2-8. CICS Region and the DSA CI207.0

Notes:

- A characteristic of CICS — from its first version — is its **single address space design**: The CICS system code and all applications are operating in **one** z/OS address space or region.
- Usually, most of CICS's private region will be used for storage areas where CICS management modules and application programs are loaded, and from where working storage both for system and application usage is taken.
  These areas are dynamically managed by CICS, and are called **Dynamic Storage Areas** (DSA).
- You specify the sizes of some areas; CICS derives others from the z/OS region size or other parameters.
- The visual above represents just the major components of the CICS TS for z/OS address space.
  The Storage Management unit will give more accurate information and explain how to estimate the region and DSA sizes your system requires.

2.3 CICS Integrity Services

Transactional Processing Characteristics

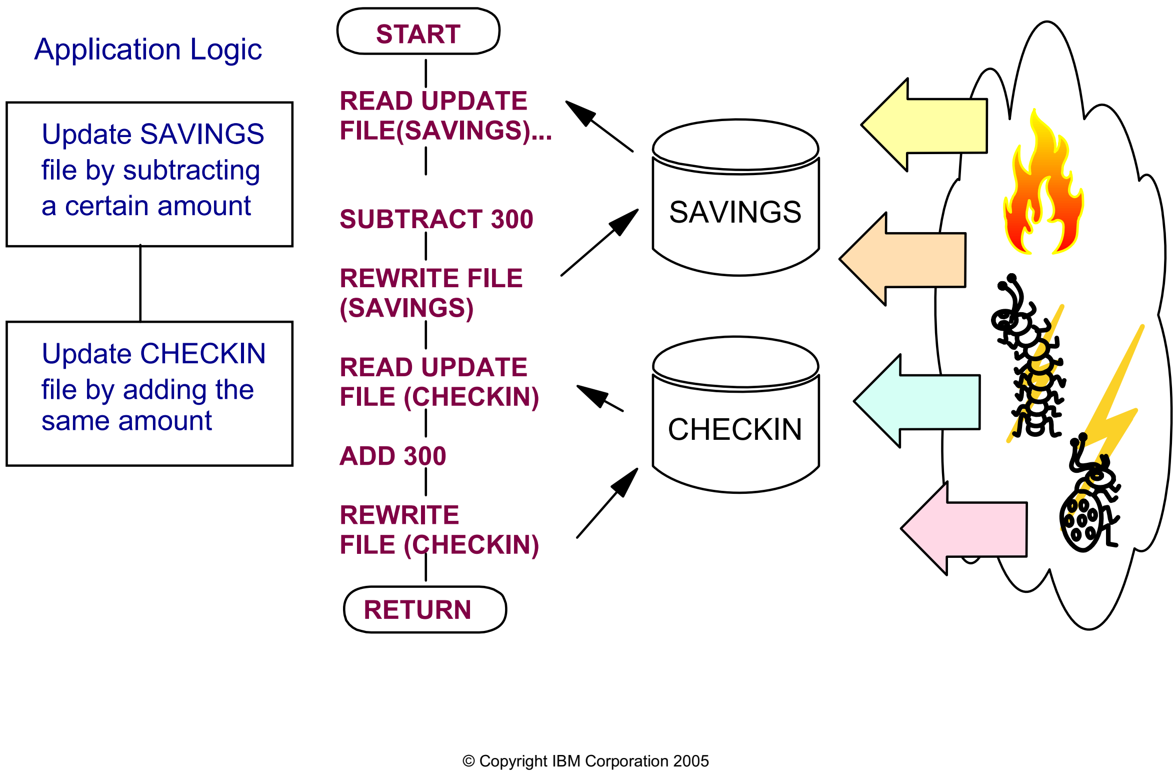


Figure 2-9. Transactional Processing Characteristics CI207.0

Notes:

- Commercial IT processing is always about accessing and manipulating business data. The main concern thereby is that data remains “clean” at any time:
  - Any changes made to the data must be saved and be consistent, as requested by the application or user;
  - Data must not be lost.
- From these fundamental needs, the **unit-of-work** (UOW) concept of transactional data processing arose, which is introduced here by a simple sample, where a bank customer wants to transfer funds from savings to checking.
  - A unit of work, which in many cases corresponds to a transaction, starts from a consistent state of data, and will change some of this data during execution.
  - CICS, as the transaction monitoring instance, has to make sure that the data which is left behind after the unit of work has terminated is consistent again.
  - Consistency of data would be threatened, for example, if, say, right after the first `REWRITE,` either the user program or even the CICS TS address space serving this

transaction were to fail or abend.

Various kinds of unscheduled interruptions may occur, caused by

- Program abends
- System abends
- Physical factors (water damage, fire, and power failure)
- Operator errors.

- If this happened, the user data would be **inconsistent** and the failures could result in the customer losing $300...
- Another threat to user data may result from multiple users or applications updating the same data resource in an uncontrolled way.

- CICS, as the transaction management system, has to ensure what is summarized by the acronym **A.C.I.D.** which has been widely spread in the transaction management world for a number of years:

Atomicity Actions (in other words, data changes) performed by a transaction have to be done either all or none.

Consistency Transactions may only work on consistent data, which means that only data that is valid and logically consistent with other data may be accessible to applications.

Isolation Actions performed by two or more transactions must be isolated from each other in a way such that each transaction may only base its actions on consistent data.

Durability Once a transaction has terminated regularly, changes of data performed by this transaction are valid and will not be withdrawn or lost.

Unit of Work Concept and Terms

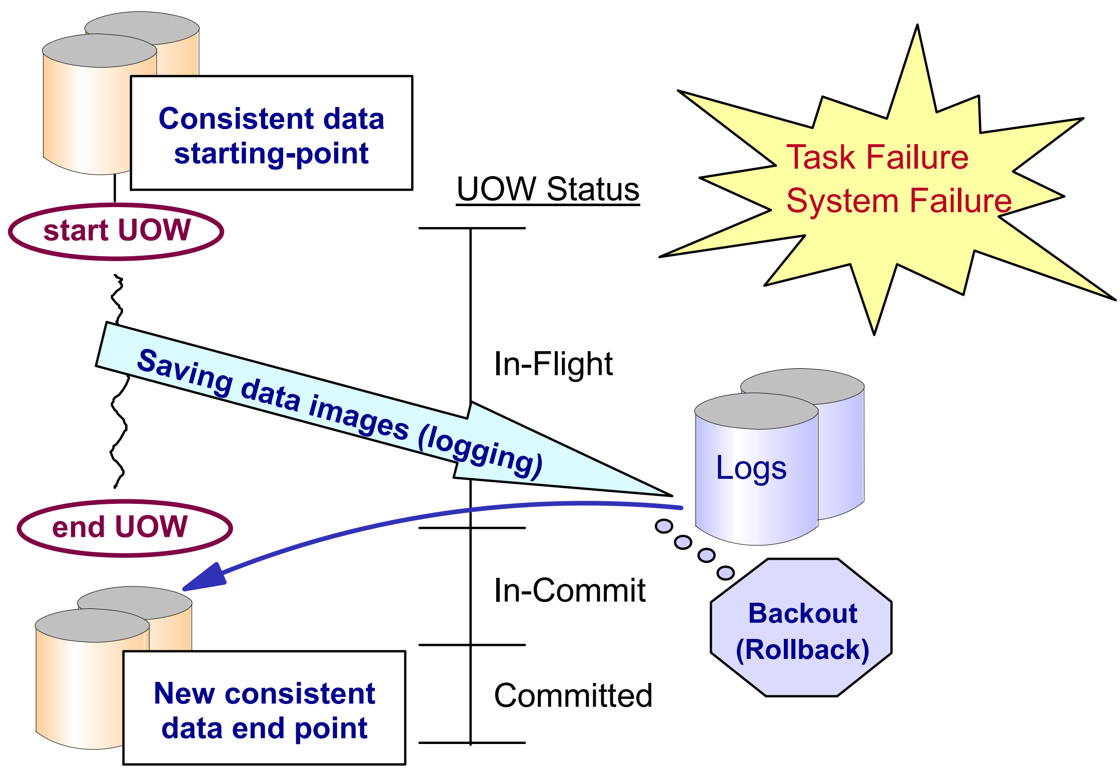


Figure 2-10. Unit of Work Concept and Terms CI207.0

Notes:

- A **unit of work (UOW)** starts when you make the first request that could change a recoverable resource, and ends when a `SYNCPOINT` command is issued or when the application terminates.
- A task is considered **in-flight** until its UOW completes.
- If a task fails while in-flight, because of a **task failure** or a **system failure**, CICS has to reset all data to the state it had when the UOW started. This process is called **backout** or **rollback**.
  - In order to be able to perform this backout, CICS has to make a copy of the data that is to be updated *before* any change is applied to it.
  - This copy is called **before image** and the action of saving this data is named **before image logging**.
    Before image logging data is written to the CICS TS system log, which can be located:
    - Within a log structure of the z/OS coupling facility
    - On DASD data sets managed by SMS.

- Backout for task failures is performed immediately, as part of the task's unit of work. In this special case, the new state of data is identical to the old one.
- Backout after a system failure is deferred until the failed CICS region is restarted again.

- CICS resources that may be protected in this way have to be defined to CICS as **recoverable resources**. They may be of the following types:
  - Files (all types supported by CICS file control)
  - Auxiliary temporary storage queues
  - Intrapartition transient data queues

  You invoke integrity service for these resources through specific options within the resource descriptions of these objects.

Enqueue Ensures Isolation

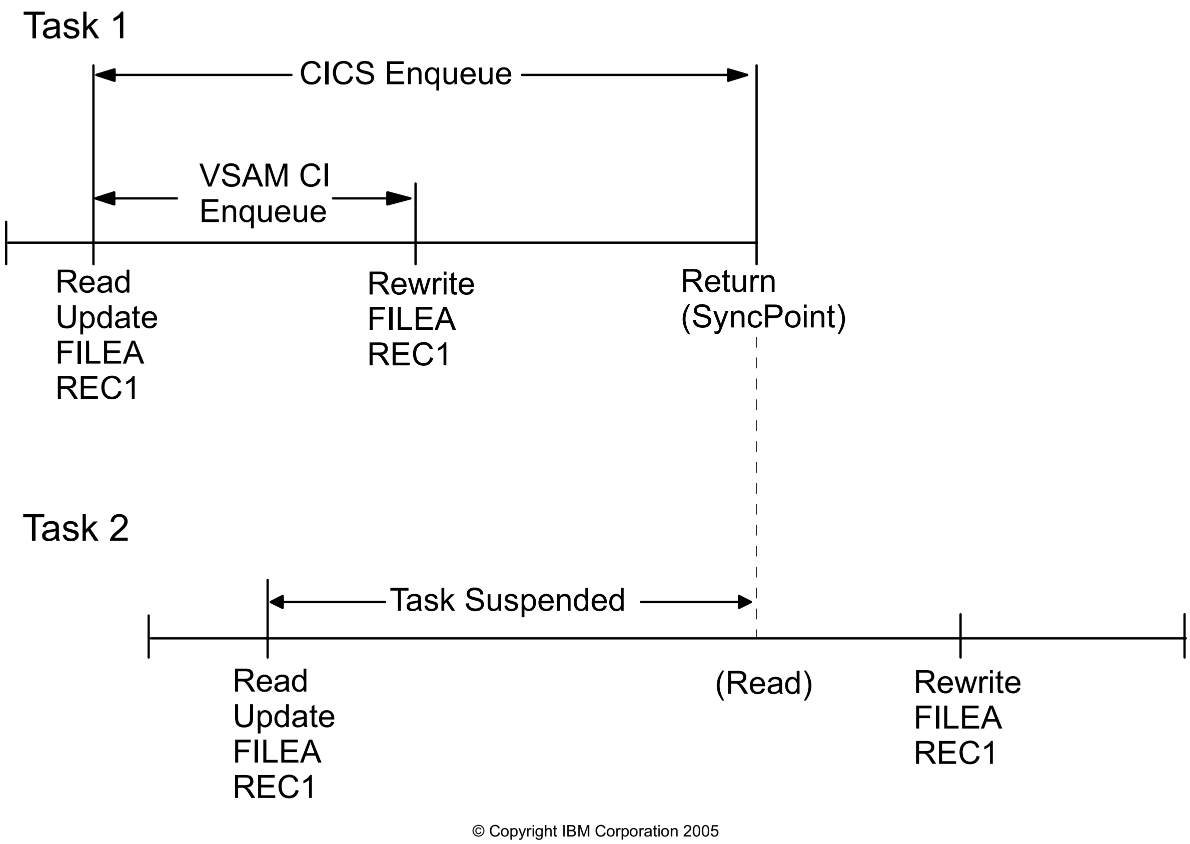


Figure 2-11. Enqueue Ensures Isolation CI207.0

Notes:

- If two tasks attempt to update the same recoverable resource, CICS provides exclusive control until the first task ends the UOW by means of an internal enqueue mechanism which is implemented by the Enqueue domain (EN).
  The second task is suspended if it tries to access the resource.
- This technique ensures that abnormal termination of Task 2 will not cause an inadvertent backout of the update from Task 1.
- Enqueue/dequeue services may be requested by applications for their business logic purposes by means of EXEC CICS ENQ / DEQ commands.
  CICS TS 1.3 extended the scope of application enqueues to the sysplex in which CICS regions are running.

File Recovery Support When Data Is Lost

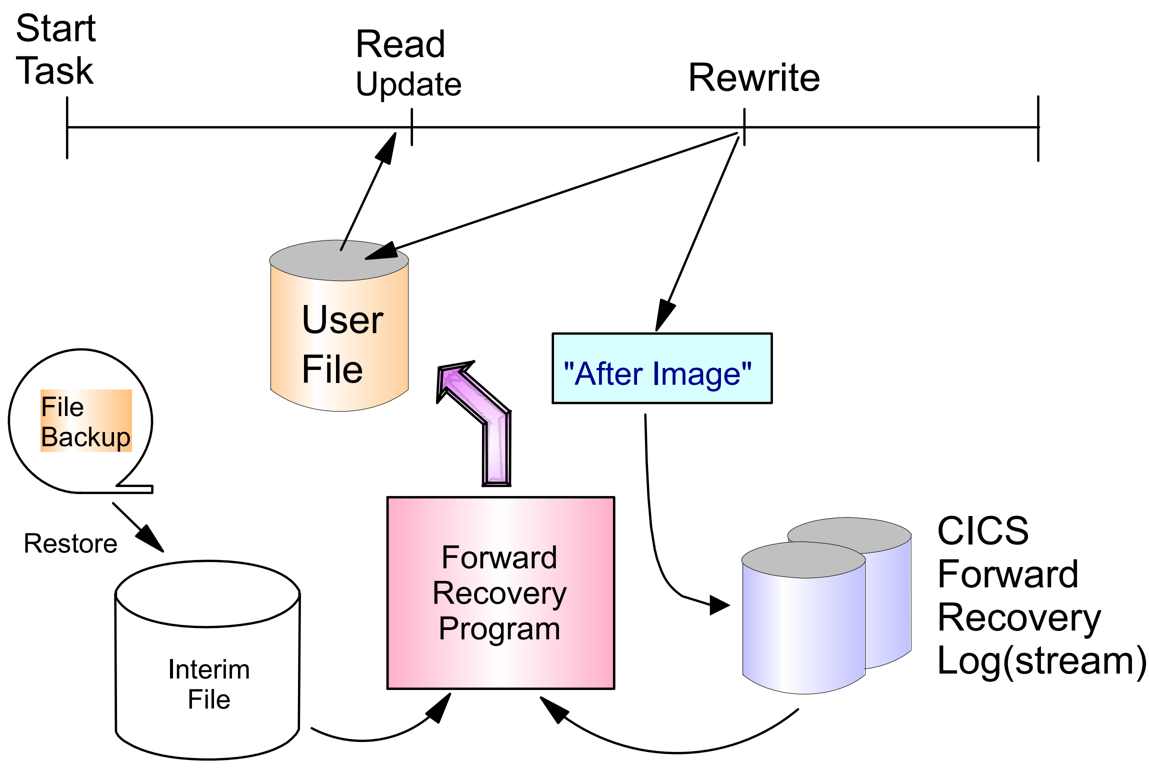


Figure 2-12. File Recovery Support When Data Is Lost CI207.0

Notes:

Imagine users making 1,000 changes to a file during the first hour of online service, then the file is destroyed (for whatever reason) and the data is no longer accessible. All changes would be lost if we didn't prepare for such a situation.

- To provide a way to recover committed updates, you can request that CICS records **after images** (the way the record looks after an update) to a log data destination, which is called the **Forward Recovery Log**.

  Again, this data may physically be written to Coupling Facility (CF) log structure storage, or DASD data sets, if no CF is available.

- To bring the file up to date, you need three things:
  - A backup copy of the file
  - All changes to the file since the backup was taken
  - A program to perform forward recovery.

- CICS provides the first two items. **You must provide the recovery program!**
  - Version 3.2, or above, of CICS VSAM Recovery (CICSVR) for z/OS and OS/390 (product number 5655-H91) may perform this role. Note that CICSVR is not an element of CICS Transaction Server.

CICS TS Logging Overview

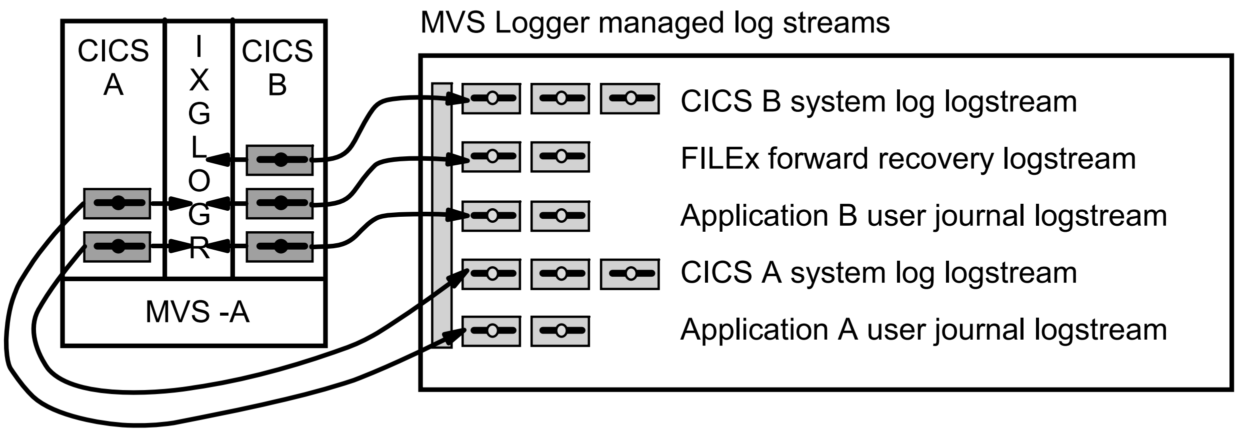


Figure 2-13. CICS TS Logging Overview CI207.0

Notes:

- Instead of doing the logging itself, the CICS Log Manager domain uses the services of the z/OS system logger, which provides an application programming interface to access records on a log stream.
- Each CICS region's system log, forward recovery logs, auto journals, and user journals map into specific log streams.
- Before images of completed transactions are deleted from the system log streams.
- Log streams are defined in structures within the coupling facility or as DASD-only log streams.

2.4 Transaction Server for z/OS Elements

CICS TS for z/OS Packaging

- CICS Base Element (V6R4)
- CICSPlex System Manager (V3R1)
- Application Migration Aid (CICS/AMA)
- REXX Development System and Runtime Facility
- CICS Integrator Adapter for z/OS V1.2
- WebSphere Application Server V5
- WebSphere Studio Enterprise Developer (WSED) V5

Figure 2-14. CICS TS for z/OS Packaging CI207.0

Notes:

All elements above are shipped with CICS TS V3.1.

- The **Base Product** is a version of the follow-up product of CICS for MVS Version 4, with significant enhancements in transaction management.
- **CICSPlex SM** (CPSM) allows you to manage a group of regions as if they were a single entity. CPSM provides a single point of control from which you enter commands that affect all CICSPlex-supported regions in the enterprise from a single TSO terminal or Web browser.
- Two of the older elements of CICS TS for z/OS continue to be available as stand-alone products as well, with their own product number, as shown above:
  - **CICS/AMA** (Application Migration Aid) is a utility to assist you in migrating macro language programs to command language. It executes in a CICS Version 1 or 2 system.
  - **REXX for CICS** allows application programs written in the REXX language to be developed and executed under CICS.

- **CICS Integrator Adapter for z/OS V1.2** is the runtime component of MQSeries Integrator Agent for CICS, provided as an element in CICS TS V3 and renamed to CICS Integrator Adapter for z/OS V1.2.
- Starting with Version 2 Release 3, the CICS TS for z/OS package includes IBM WebSphere products in order to support the development and execution of Java enterprise applications, namely:
  - A restricted licence of **WebSphere Application Server V5** for various platforms, such as IBM iSeries, Solaris, Windows 2000, and others (but not for z/OS), which provides a runtime environment for web applications that may cooperate with appropriate newly-developed CICS application components.
  - An unrestricted single user licence of **WebSphere Studio Enterprise Developer (WSED) V5**, which offers an Integrated Development Environment for the creation of dynamic web applications, including support for Java 2 Enterprise Edition (J2EE) XML and Web services technologies. This is to be installed on a workstation as well.

  Note that the installation and use of these products is not documented by CICS TS manuals; you just get the media packages (a couple of CD-ROMs) as part of the CICS TS for z/OS package.

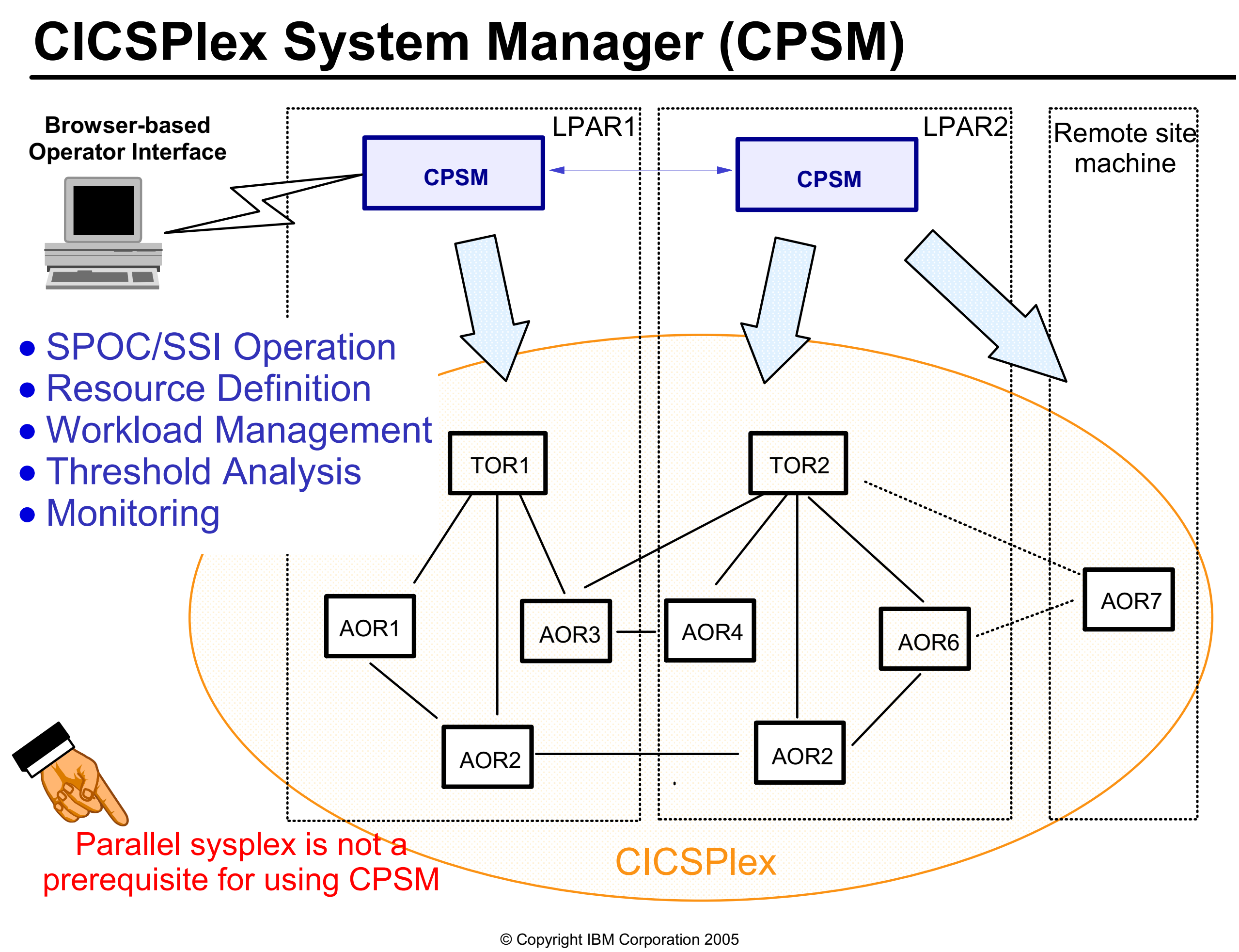


Figure 2-15. CICSPlex System Manager (CPSM) CI207.0

Notes:

- CICSPlex System Manager (CPSM) is by far the “mightiest” of the additional elements of CICS TS. It allows for enterprise-wide CICS system management by providing, among others, the following functions:
  - Single point of control (SPOC) for any set of CICS regions
  - Single system image (SSI) operating (“super-CEMT”)
  - Centralized management of CICS application resources
  - Automated monitoring
  - Control of dynamic transaction routing (workload balancing) between any number of CICS regions running in parallel
- Although CPSM ships as part of CICS Transaction Server, it has to be installed separately, and is made up of at least two address spaces within every LPAR in which one or more CICS regions are running.
- Note that z/OS parallel sysplex in not a prerequisite for using CPSM, although sysplex environments are likely to benefit most from it, as there may be a big number of CICS regions that may be difficult to control otherwise.

Latest Product Enhancements

- CICS TS Version 1
  - CICS Transaction Server for OS/390 V1.3 was launched with the clear and simple message; "Evolution not Revolution"
- CICS TS Version 2
  - EJB Support including CORBA
  - Enhancements to the DB2 attachment
  - ECI over TCP/IP
  - SOAP for CICS (optional feature)
- CICS TS Version 3
  - Web Services support
  - Enhancements to HTTP support
  - Enhancements to the Open Transaction Environment
  - Enhancements to C/C++ support
  - New inter-program data transfer (32-KB COMMAREA limit)
  - Codepage conversion enhancements

Figure 2-16. Latest Product Enhancements CI207.0

Notes:

This visual shows an incomplete list of the functional enhancements that have been made to CICS TS during the last few years, with references to the appropriate releases.

The main titles for enhancements and new functions were:

- Parallel sysplex support
- System management
- Application support and solution enablement
- e-business enablement for network computing
- Enterprise Java Beans and Java Virtual Machine support
- Enterprise management
- Web services support

We will refer to a number of these functions during the following units. Note, however, that Java-related functions are completely covered by course CI21, *CICS TS Java Enablement*.

CICS TS for z/OS Overview (Summary)

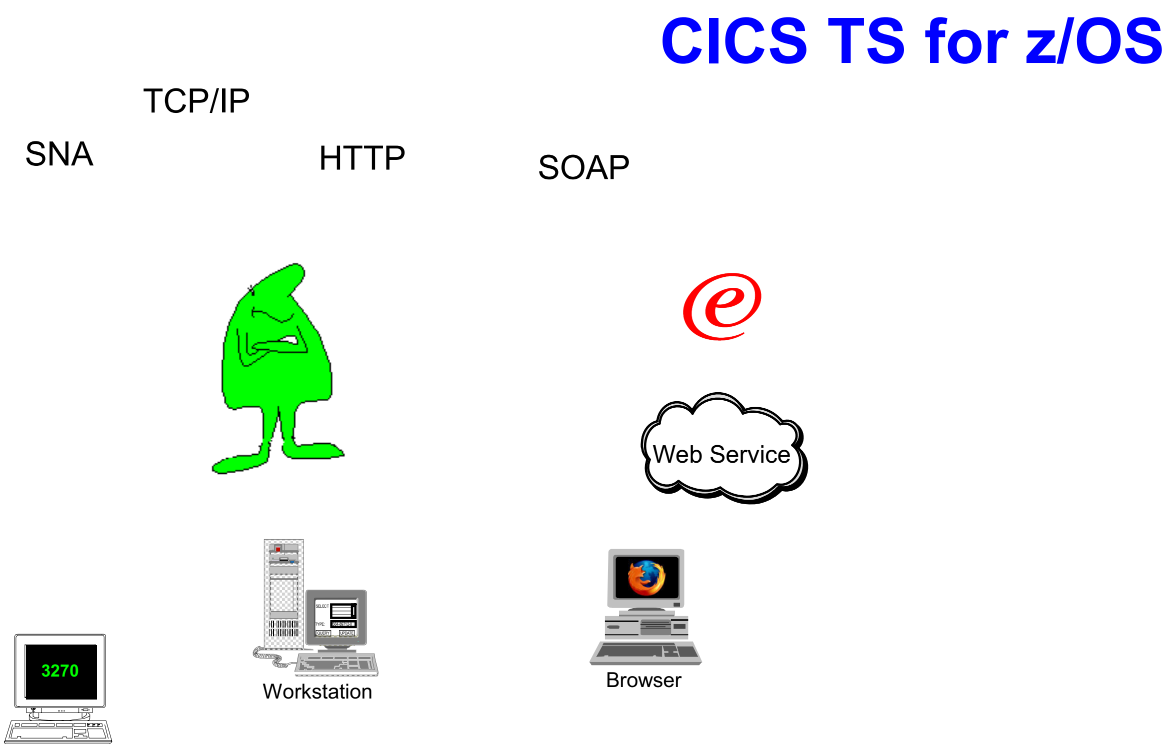


Figure 2-17. CICS TS for z/OS Overview (Summary) CI207.0

Notes:

- We often refer to CICS as a Data Base/Data Communication (DB/DC) product, since it allows communication devices to access databases and files.
- In modern terms, CICS is called an **application server**, as it delivers various kinds of services to applications running in a single region as well as in a distributed Client/Server environment.
- Users generally initiate transactions from a terminal, but increasingly, transactions may be generated by some devices and facilities connected to CICS. In any case, a transaction is a request for work and must be predefined to CICS TS.
- CICS uses its own task management to support multiple users.
- You define resources for your users' applications. CICS stores these definitions in tables managed by the CICS management modules.
- All applications that execute within an instance (region) of CICS for z/OS operate in the same address space that is also used by system programs. CICS manages these **dynamic storage areas**.

- An open-ended design allows maximum flexibility in tailoring CICS TS to your present and future needs.
- The CICS TS for z/OS package contains elements that provide additional support for system management, sysplex exploitation, and network connectivity, including Internet access.

Unit Summary

Having completed this unit, you should be able to:

- Identify the role of CICS TS in different approaches of Information Technology (IT) infrastructures
- Identify the major CICS TS for z/OS functional components
- Describe the major areas of a CICS TS address space and the parameters that influence the sizes of these areas
- Describe, at a high level, the flow of a typical CICS task
- Identify the most important system resources used by a task

Figure 2-18. Unit Summary CI207.0

Notes:

Unit 3. Installation and Verification

What This Unit Is About

This unit gives an overview of how installation of CICS Transaction Server for z/OS is done using the means supplied by the software delivered. Besides, you learn about the pre- and co-requisites that have to be considered before and during the installation.

The system-supplied installation verification aids are introduced briefly.

What You Should Be Able to Do

After completing this unit, you should be able to:

- Name the most important hardware and software requirements to run CICS TS
- Describe the flow of steps necessary to install CICS TS
- Add CICS procedures to your procedure library
- Integrate CICS into the operating system
- Create CICS system and test data sets
- Verify CICS installation with supplied procedures.

How You Will Check Your Progress

- Machine exercise lab.
  You will create your own set of CICS system data sets by use of the CICS-supplied DFHISTAR job generator mechanism.

References

CICS Transaction Server for z/OS V3.1 Program Directory, GI10-2586

CICS Transaction Server for z/OS Version 3.1 Installation Guide, GC34-6426

Unit Objectives

After completing this unit, you should be able to:

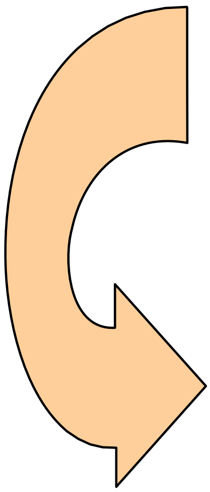
- Name the most important hardware and software requirements to run CICS TS
- Describe the flow of steps necessary to install CICS TS
- Add CICS procedures to your procedure library
- Integrate CICS into the operating system
- Create CICS system and test data sets
- Verify CICS installation with supplied procedures

Figure 3-1. Unit Objectives CI207.0

Notes:

3.1 Basic Installation of CICS TS for z/OS

Pre-Installation Planning



- Release Guide
- Migration Guide
- Program Directory
- Review hardware / software requirements
- Support Center

Figure 3-2. Pre-Installation Planning CI207.0

Notes:

- Comprehensive information on the hardware and software requirements to install and run CICS TS for z/OS is provided by the books mentioned above.
- These are the complete titles and form numbers of the current versions:
  - *CICS Transaction Server for z/OS Version 3.1 Release Guide, GC34-6421*
  - *CICS Transaction Server for z/OS Migration Guide, GC34-642x*
  - *CICS Transaction Server for z/OS V3.1 Program Directory, GI10-2586*
- The Program Directory document is delivered with the product. It contains current information about installing CICS, and should be reviewed before proceeding further.
- To get the most recent information, contact the IBM Support Center. They should be contacted twice: after receiving CICS from IBM, and again immediately before the installation.

Delivery and Installation Methods

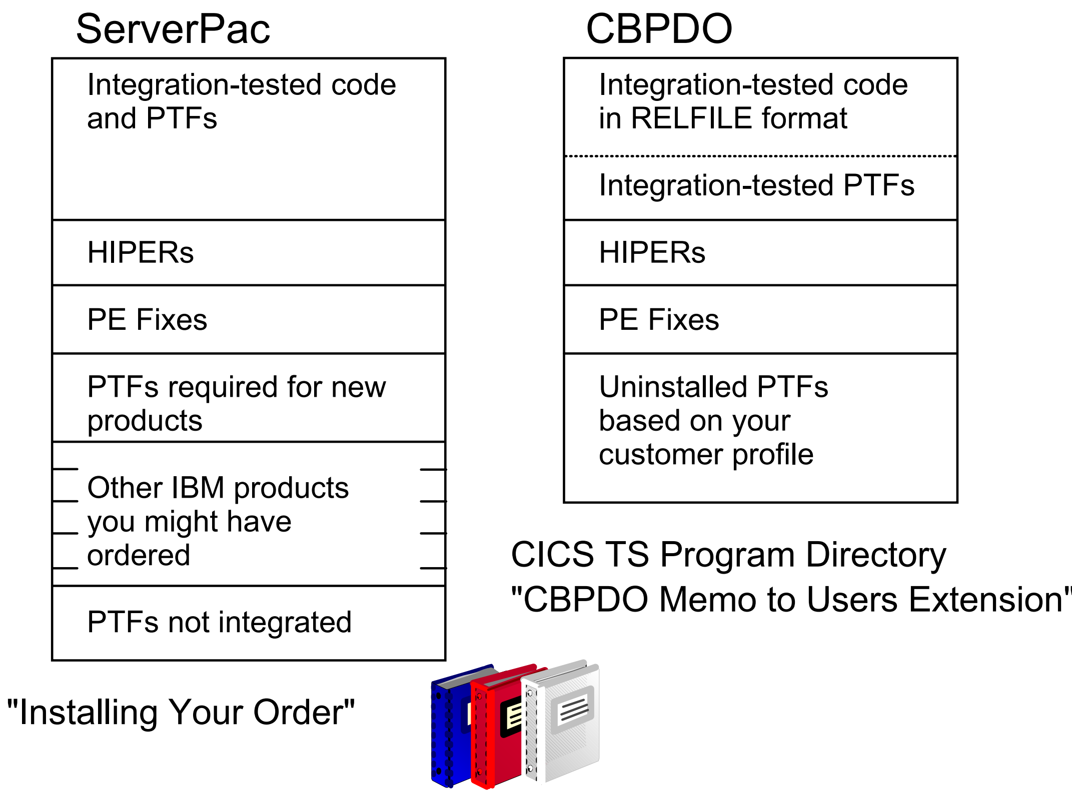


Figure 3-3. Delivery and Installation Methods CI207.0

Notes:

- CICS TS may be installed based on either the z/OS **ServerPac** or the MVS Custom-Built Product Delivery Offering **CBPDO** method of delivery.
  - The ServerPac method is based on an ISPF-guided dialog, through which all CICS product libraries, including the SMP/E software inventory data sets, are unloaded to your volumes, by names that you specify.
    Usually, other IBM products are delivered and installed at the same time by use of a ServerPac.
  - The CBPDO method delivers one SMP/E relfile tape that contains all CICS TS base elements. As guided by the *Program Directory*, you have to use SMP/E to build up your CICS TS target product libraries and the SMP/E environment which enables further maintenance of the CICS TS software.
  - For both methods, specific installation support information is provided as hardcopies.
- All post-installation tasks of setting up specific CICS regions, verifying, and applying maintenance are discussed in the *CICS Transaction Server for z/OS Version 3.1 Installation Guide, GC34-6426*.

Hardware Requirements

- **Any model of the System/390 processor family of servers:**
  - Must support z/OS V1R4
- **Coupling Facility is required for the following functions:**
  - Share VSAM files, based on VSAM RLS
  - Shared Coupling Facility Storage Pools (DT and TS)
  - Coupling Facility Logging

Figure 3-4. Hardware Requirements CI207.0

Notes:

- The minimum operating system level for CICS TS Version 3 Release 1 is z/OS Version 1 Release 4. This requirement is checked by CICS initialization code.
- A coupling facility is not required for the CICS system log as DASD only is supported. But it is mandatory to support VSAM RLS, TS shared and Data Table in coupling facility.
- More detailed specifications are contained in the planning material introduced previously.

Software Requirements

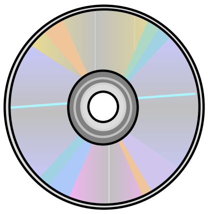
- **Minimum requirements for CICS TS for z/OS base functions:**
  - z/OS V1R4 with JES2 or JES3
  - Herein, the following elements are included that are required for installing and running CICS TS
    - DFSMS/MVS (previously DFP)
    - Language Environment
    - SNA and IP component of eNetwork Communications Server (previously VTAM and TCP/IP)
    - Security Server (previously RACF)
    - z/OS UNIX System Services
    - SMP/E for z/OS and OS/390 Version 3 release 1

Figure 3-5. Software Requirements CI207.0

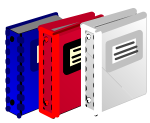
Notes:

- A detailed discussion of hardware and software prerequisites is contained in *CICS Transaction Server for z/OS V3.1 Program Directory, GI10-2586* and the appropriate Release Guides.
- The latest information about required maintenance to z/OS or other system software products is given with the program directory or Memo To User that is delivered with the CICS TS for z/OS software.

CICS TS for z/OS Publications



- **CICS TS Information Center**
  Complete set of books plus additional material - CD-ROM delivered with the product
- **Transaction Processing and Data Collection Kit**
  All unlicensed product books - separately orderable



- **Entitlement Hardcopy Books**
  A small set of essential publications - delivered automatically with the product

- **Orderable Hardcopy Books**
  Separately orderable hardcopy books
- **Online library**
  Unlicensed books in PDF format - downloadable from
  `www.ibm.com/software/htp/cics/tserver/v31/library/`

Figure 3-6. CICS TS for z/OS Publications CI207.0

Notes:

- CICS TS documentation does not come as hardcopy manuals with the product, but is provided on the CICS TS Information Center CD, a copy of which is shipped with the product automatically. Additional copies may be ordered from IBM free of charge. It is also downloadable from the Internet.
  - The CICS TS Information Center is launched from a web browser. It is a graphical user interface providing access to an HTML representation of the total CICS library, and to a PDF file for each of the books.
    It runs on Windows XP, Windows 2000, and AIX 5.1+, and requires a browser supporting HTML 4.0, such as Internet Explorer 6 or Mozilla 1.4.
  - In CICS TS V3.1, the Information Center is powered by Eclipse technology. It consists of an Eclipse Help System, with the information for CICS TS as a plug-in.
  - Only a version without a licensed book is available on the Internet.
- Some installation publications needed for planning, installation, and initial setup continue to be delivered as hardcopies with the product.

- Other hardcopy books are available in a set, ordered by a single, chargeable order number available to customers when they order the product.
- The licensed books are available, to license holders only, through feature numbers.

Installation Process Overview

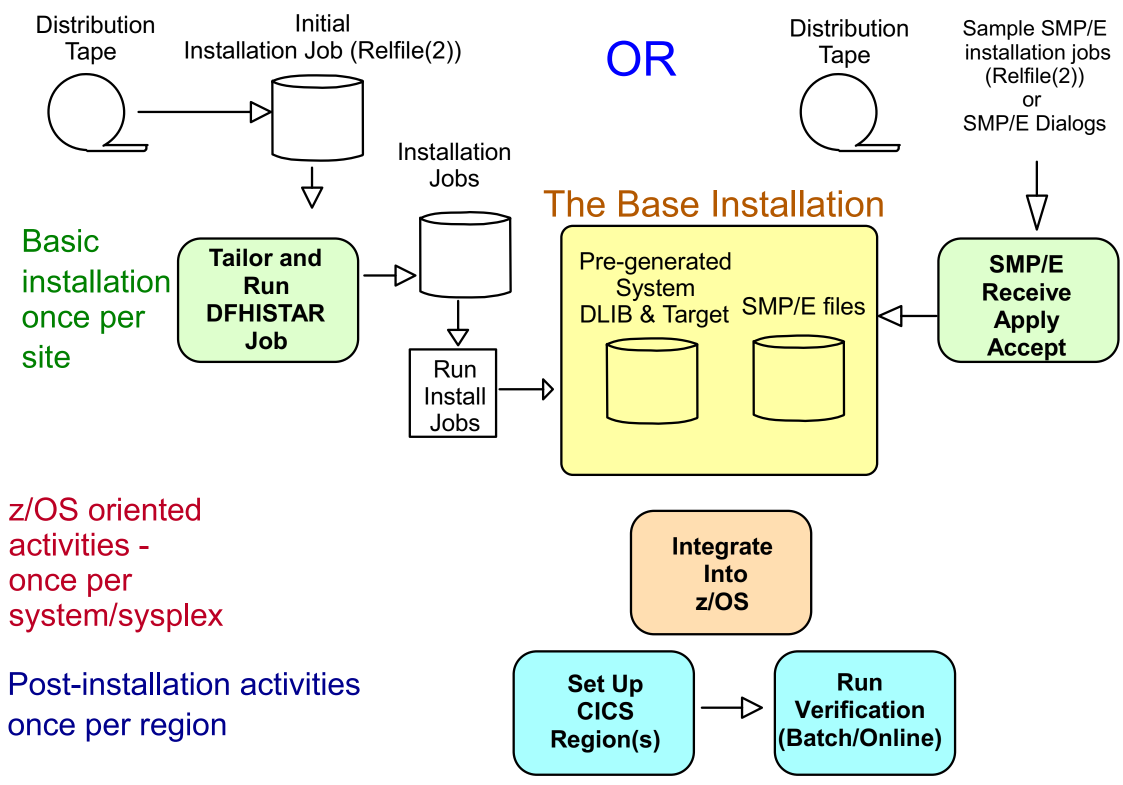


Figure 3-7. Installation Process Overview CI207.0

Notes:

- This is an overview of the maximum number of steps that are necessary to install CICS.
- The whole procedure is applicable to the **CBPDO** delivery method only.
  For this method, the *CICS Transaction Server for z/OS V3.1 Program Directory, GI10-2586*, together with your customized issue of the *CBPDO Memo to Users Extension*, describes the installation process for the current version of CICS, whereas the post-installation tasks are documented in the *CICS Transaction Server for z/OS Version 3.1 Installation Guide, GC34-6426*.
- IBM recommends use of the **ServerPac** method, in which you receive a series of tapes, each in IEBCOPY dump-by-data set format containing a complete generated CICS TS system. This consists of distribution and target libraries, consolidated software inventory (CSI), and other SMP/E libraries already generated. CICS TS elements and their service are integrated into distribution and target libraries.
  Installation activity is driven completely by the ServerPac ISPF dialog, and you can start with the post-installation activities thereafter.

DFHISTAR JCL Generator Facility

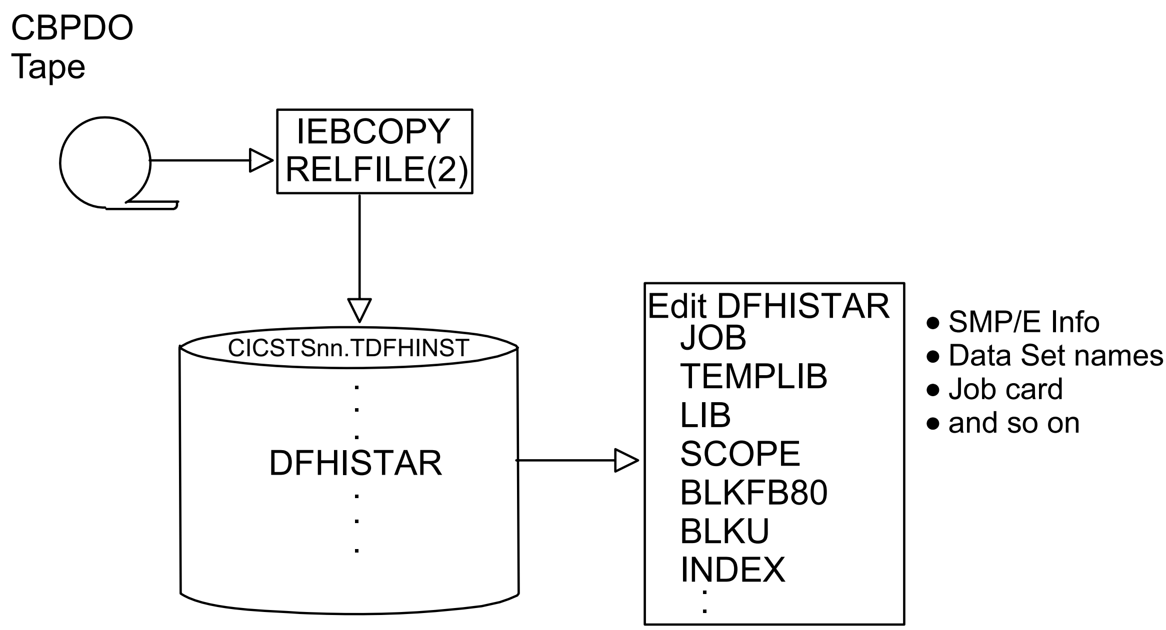


Figure 3-8. DFHISTAR JCL Generator Facility CI207.0

Notes:

Here you start when you are a CBPDO user.

- The *CICS Transaction Server for z/OS V3.1 Program Directory, GI10-2586*, gives a sample IEBCOPY job that you can customize to copy RELFILE(2) from the CICS TS CBPDO tape.
- By using the CICS TS job generator provided with member **DFHISTAR** of the library accessed, you generate a single set of install jobs:
  - You have to edit the DFHISTAR member in order to specify a number of parameters that are used to construct the JCL generated.
  - All parameters are described in detail in the *Program Directory*.
  - One of the most significant parameters you are asked to specify is the high-level qualifier for the data sets into which the jobs install the product. If you use the default, the CICS product data sets will be named `CICSTS31.CICS.SDFH*`.
- DFHISTAR is used for some post-installation tasks as well, for example creating the data sets required for a specific CICS region, regardless of the delivery and installation method.

Output of DFHISTAR

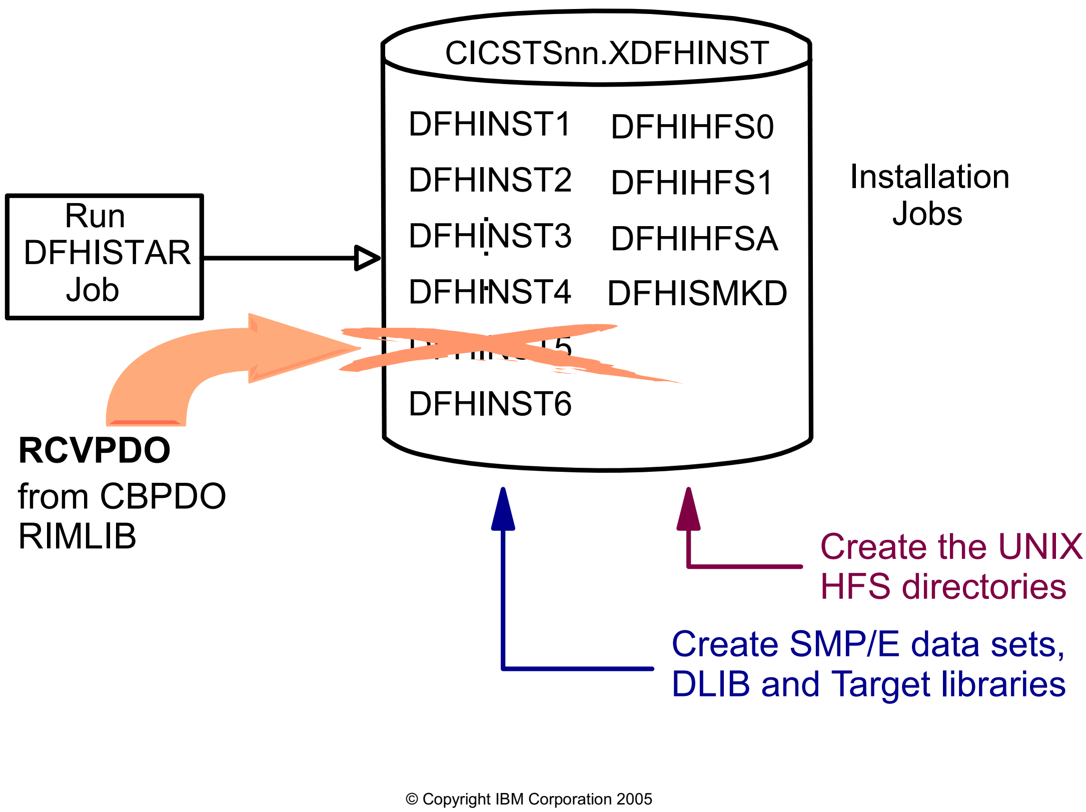


Figure 3-9. Output of DFHISTAR CI207.0

Notes:

- On completion of the DFHISTAR job, the output library contains the jobs generated from DFHISTAR parameters. These have to be run in order to complete the installation.
- Jobs DFHINST1 to 6 are SMP/E-oriented and perform the SMP/E-controlled installation of CICS TS to local DASD. The SMP/E RECEIVE job (DFHINST5) has to replaced by a CBPDO-provided job named **RCVPRO**.
- Additional jobs are generated by the DFHISTAR script that creates the HFS directories in the z/OS UNIX environment, needed to install CICS support for *VisualAge for Java, Enterprise Edition for OS/390*, and *IIOP inbound to Java* applications.
  - Variables for these areas, such as HFS directory names, are specified by DFHISTAR parameters as well.
  - To install these components (FMID JCI640D), z/OS UNIX must be running in full-function mode on the MVS image being used to install the product.
- The install jobs are designed to install the complete program product, with all elements of CICS TS, which means all FMIDs. *CICS Transaction Server for z/OS V3.1 Program*

Directory, GI10-2586, states: *Do not* modify the install jobs to exclude any of the FMIDs, otherwise you may find that you are unable to apply CICS TS service.

- A job named **DFHIJVMJ** is created by DFHISTAR, in order to customize CICS-supplied sample versions of JVM profiles that are needed for the execution of Java programs within CICS.
- For other prerequisites and details for the installation of these components, refer to the installation supporting documents mentioned earlier.

CICS TS for z/OS Libraries

SDFHINST	Installation jobs and procedures
SDFHPROC	General procedures
SDFHSAMP	Sample programs, definitions, and so on
SDFHC370 SDFHCOB SDFHPL1 SDFHMAC	CICS structures and constants for use by applications programs - for named languages
SDFHSDCK	Side deck control input for C++ programs
SDFHLOAD SDFHAUTH SDFHEXCI SDFHLINK SDFHLPA	Executable program libraries
SDFHJLOAD SDFJLOD1	Libraries to be loaded to HFS for Java and IIOP support
SDFHDLL1	HTTP Server plug-in
SDFHSRC	Source Code (AP domain only)

Figure 3-10. CICS TS for z/OS Libraries CI207.0

Notes:

- These are the most important libraries that make up the CICS TS base product.
- Additional libraries will be created by the installation process covering the following functions:
  - The Message Editing Utility (MEU), which is supported by a set of libraries that have to be allocated on the ISPF level.
  - Two libraries containing data to be loaded to the sample application files.
  - A library from which to initially load the Online Messages and Codes file.
- If you install other elements of CICS TS, more (target) libraries will be there.
  - The CICSPlex SM libraries are named `CICSTS31.CPSM.SEYU*`.
  - The REXX for CICS libraries are named `CICSTS31.REXX.SCIC*`.
- CICS TS libraries can reside on 3380 or 3390 DASD. The space required by the CICS TS installation is about 2,000 tracks (3390) for the SMP/E data sets and about 3,000 tracks each for the distribution and target libraries, plus 40 cylinders of HFS space in z/OS UNIX.

Program Libraries

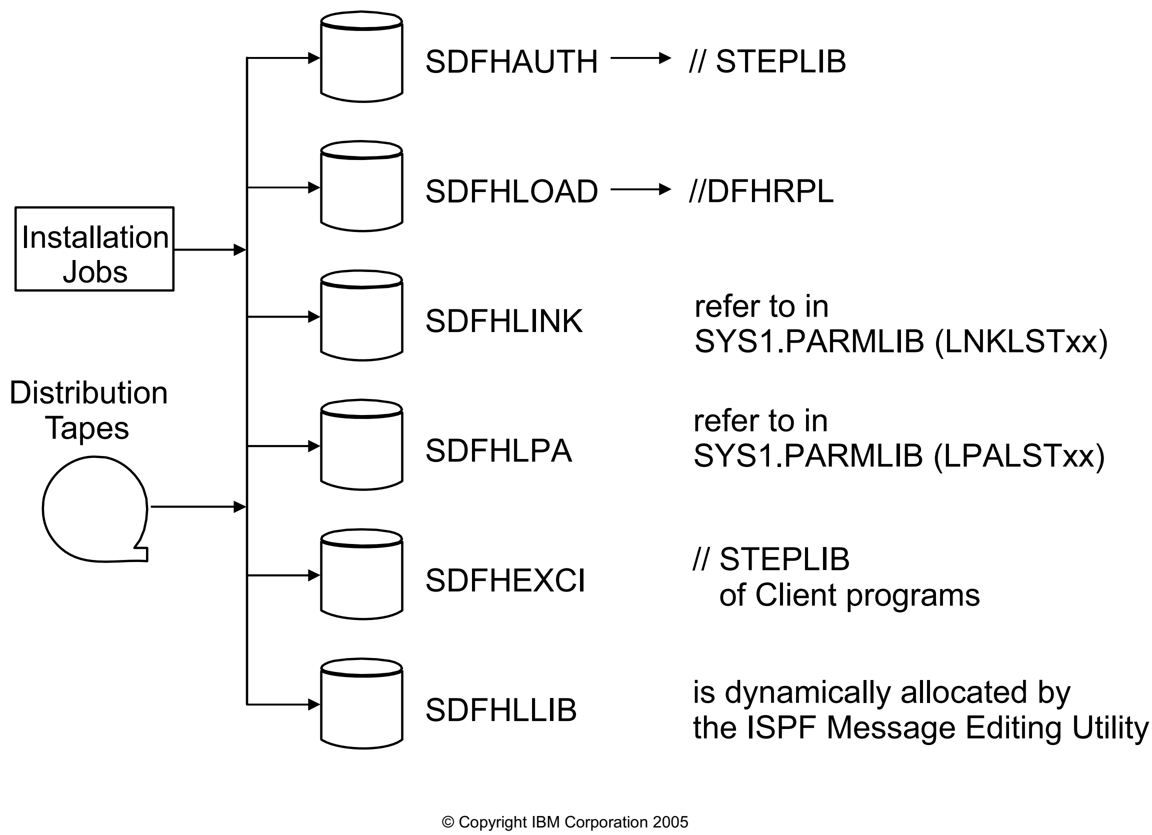


Figure 3-11. Program Libraries CI207.0

Notes:

These are the six load libraries containing executable programs that are required:

- To run CICS TS for z/OS (SDFHAUTH, SDFHLOAD, SDFHLINK, SDFHLPA)
- To make use of the External CICS Interface from a z/OS-based program
- To support the Message Editing Utility.

The visual does not show the PDS/E and HFS libraries that are used to support CICS Java-based programs.

CICS Procedures

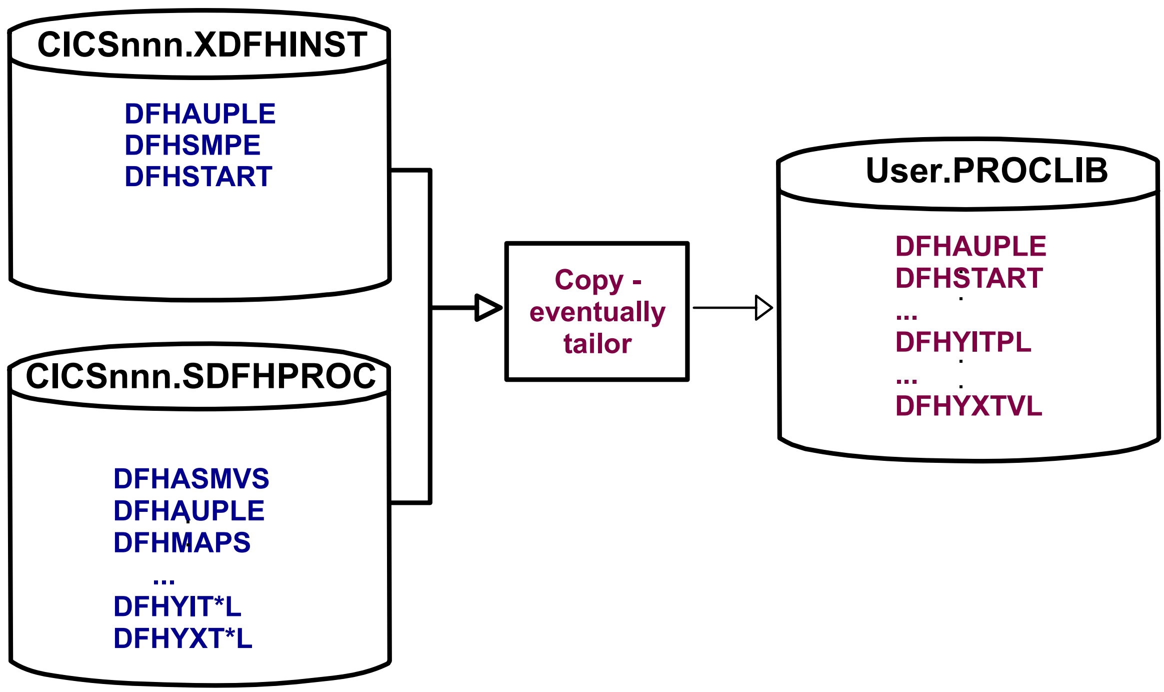


Figure 3-12. CICS Procedures CI207.0

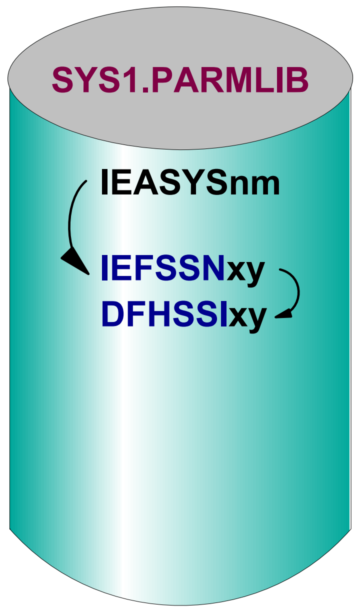
Notes:

- Procedures in XDFHINST are created by DFHISTAR, and should already meet your installation standards, as you specified your resource names in the DFHISTAR input control parameters.
- The procedures supplied in SDFHPROC are mainly for application preparation:
  - Procedures named DFHYIT*L translate, compile, and link-edit application programs using the command-level interface under Language Environment.
  - Procedures named DFHYBT*L do the same for programs using EXEC DLI commands.
  - Procedures named DFHYXT*L are for programs using the external CICS TS interface (EXCI).
  - For all members, the “*” character is “D” for C/370, “E” for C++, “P” for PL/I, and “V” for COBOL.
- You have to inspect these procedures and eventually must tailor them in order to have your compiler data sets accessed correctly.

- If you run a previous version or release of CICS, these procedures are likely to be there already. The variables used by the procedures may change between CICS releases, so they might have to be invoked differently.
  - Take care to ensure that the productive (“old”) procedures are still working unchanged when you activate the new ones.
  - The easiest way to ensure compatibility may be to use a separate procedure library for the new release during the test phase, and access it by use of a JCLLIB specification in the invoking JCL, or via a PROC=xx statement after defining the appropriate proclib concatenation to JES.
- CICS Transaction Server for z/OS Version 3 no longer supports pre-Language Environment compilers, as with program preparation. Accordingly, the procedures named DFHEIT*L have been dropped.
- OS/VS COBOL: As indicated in previous announcements, CICS runtime support for OS/VS COBOL (5740-CB1, 5734-CB4, and 5740-LM1) is not provided in CICS TS V3.1.
- In CICS Transaction Server for z/OS, Version 3 Release 1, OS PL/I, and C/370 runtimes are removed. Applications compiled and linked with these non-Language Environment-conforming products usually execute successfully under Language Environment in compatibility mode. It is recommended to recompile applications using the LE compilers, whenever possible.

  A very useful overview of the current support status of compilers and runtime environment is provided in “High-level Language Support”, chapter 27 of the *CICS Transaction Server for z/OS Version 3.1 Release Guide, GC34-6421*.

Base Integration With MVS (1)



Define CICS as an MVS subsystem

Specify message-formatting

Specify EXCI pipe allocation limit

Figure 3-13. Base Integration With MVS (1) CI207.0

Notes:

As CICS is operating as a formal MVS subsystem, and needs to make use of some special MVS facilities, some MVS customization is required before a CICS region can be started. These activities are required, irrespective of the particular CICS regions that will be created and started thereafter.

- All of these items are based on entries in certain members of the `SYS1.PARMLIB`, as shown in the visual.
- The “xy” suffixes used may differ per member. Entries in the IEASYSxx member point to the active members at MVS IPL time. The IEASYS suffix itself is specified with the MVS start (IPL) command.

1. **CICS has to be defined as an MVS subsystem** within member IEASSNxy.
   - The entry is release-independent and looks like this:
     `SUBSYS SUBNAME(CICS) INITRTN(DFHSSIN) INITPARM(DFHSSIyy)`
   - This DFHSSIyy member pointed to is another member in SYS1.PARMLIB where the message-formatting initialization and EXCI pipe allocation limit are defined.
   - This one entry supports the operation of any number of CICS regions.

Base Integration With MVS (2)

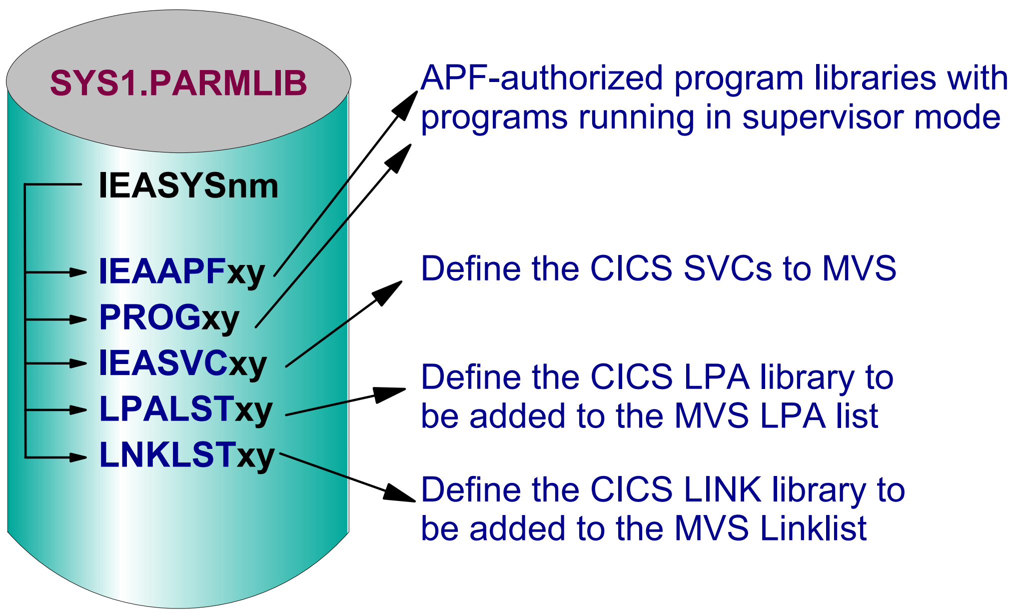


Figure 3-14. Base Integration With MVS (2) CI207.0

Notes:

2. **The following CICS libraries have to be APF-authorized**, as they contain programs that must run in supervisor mode: `SCSQAUTH, SCSQLPA, SCSQLINK.`
   - If you use new library names for CICS execution, you have to newly authorize the appropriate libraries.
   - PARMLIB member IEAAPFxy is for static APF-definitions, where changes need an MVS IPL to take effect,
   - Member(s) PROGxy are for dynamic APF-definitions that may be changed by use of the z/OS commands `SETPROG` or `SET PROG=xy`.

3. **CICS provides two SVCs** (supervisor call routines) that have to be installed in MVS: a mandatory one and an optional one.
   The appropriate entries in PARMLIB member IEAAPFxy look like this:

```
SVCPARM 216,REPLACE,TYPE(3),EPNAME(DFHCSVC)
SVCPARM 215,REPLACE,TYPE(6),EPNAME(DFHHPSVC)
```

- The first entry is for the so-called CICS SVC, which is mandatory for CICS operation. The number specified in the IEAAPFxy entry is referenced by a CICS startup parameter, 216 being the default.
  - CICS initialization contains a routine that checks for the CICS SVC being active at the latest CICS release.
  - The CICS SVC is downward-compatible with earlier releases of CICS. All currently supported releases of CICS can be run with the CICS SVC supplied with the newest release.

- The second entry shown is for the optional “High Performance SVC”. When used, this reduces the path length for some VTAM-related functions.
  - CICS startup parameters determine whether or not a particular CICS region uses this option and specify the SVC number (the default is 215).
  - This SVC is nucleus-linked, so the initial installation requires some activity on the MVS level; but actually the code has not changed for a number of CICS releases, so no actions are required if the SVC has been installed with an older version of CICS (back to CICS 1.7).

4. Modules supplied in the load library `hlq.SDFHLPA` must be loaded to the **MVS link pack area** (LPA or ELPA, respectively) for execution.
   Appropriately, this library (or one to where these modules have been copied) must be specified in the PARMLIB LPA list member.
   - All these members are downward-compatible with equally named members of previous releases of CICS.
   - A number of other CICS system modules, as well as application programs, may *optionally* be loaded to, and run from, an MVS link pack area. This is discussed later in the course.
5. Modules supplied in the load library `hlq.SDFHLINK` must be accessible through the MVS link list, because their location cannot be specified by JCL.
   Appropriately, this library (or one to where these modules have been copied) must be specified in the PARMLIB link list directory.
   - For the release-specific members, a release identification is part of the member name. The other members are downward-compatible with previous releases of CICS.
   - If multiple releases are to be run concurrently within a z/OS, the appropriate SDFHLINK libraries have to appear in the correct sequence, starting with the newest release.
6. Optionally, a program properties table (PPT) entry may be provided for CICS, which is done in PARMLIB member named SCHEDxy (not shown in the visual).
   The most important reason for doing this is to define CICS as non-swappable, for performance reasons.

- All activities are discussed in great detail in the “Setting Up the MVS Environment” part of *CICS Transaction Server for z/OS Version 3.1 Installation Guide, GC34-6426*.

3.2 Set up and Verify CICS Regions

Region Setup Tasks - Overview

Provide (basic) RACF Security Definitions — covered by unit 13

Define CICS Region to VTAM — covered by unit 5

Define the Logger Environment — covered by unit 12

- Run the Post-Installation setup jobs to create the region data sets
- Create the CICS startup job/procedure
- Run the documented Installation Verification Procedures (IVP)

shown and discussed next

Figure 3-15. Region Setup Tasks - Overview CI207.0

Notes:

The visual names the tasks to be performed with respect to one or more particular regions to be started, after the basic installation and integration tasks have been done.

Details about logging, security, and network-related definitions will be covered by the appropriate units, but take note of the following facts, as to the **minimal efforts needed** to have your first CICS TS region ready for start-up:

- Security permissions will be required at least to permit to your CICS region access to data sets and MVS resources that are RACF-protected. The userid to be permitted is the one by which the CICS region is run. Normally, these permissions are not release-specific.
- For each CICS region that is to support logons from terminals, a VTAM application minor node has to be defined. This definition is not CICS release-specific, so you may refer to a VTAM definition that served some CICS region that is not in use at this time.
- CICS may be run, for test purposes, with so-called “dummy log streams”, which means that no real logging is performed, and thus no logger resources have to be there. Of

course, no restart activities can be performed when using this mode of operation, and CICS can only be started without any reference to a previous run, which is called an “initial start”.

The DFHISTAR job generator creates a set of jobs that can be used to create the data sets needed by a particular CICS region.

CICS TS for z/OS Region Data Sets

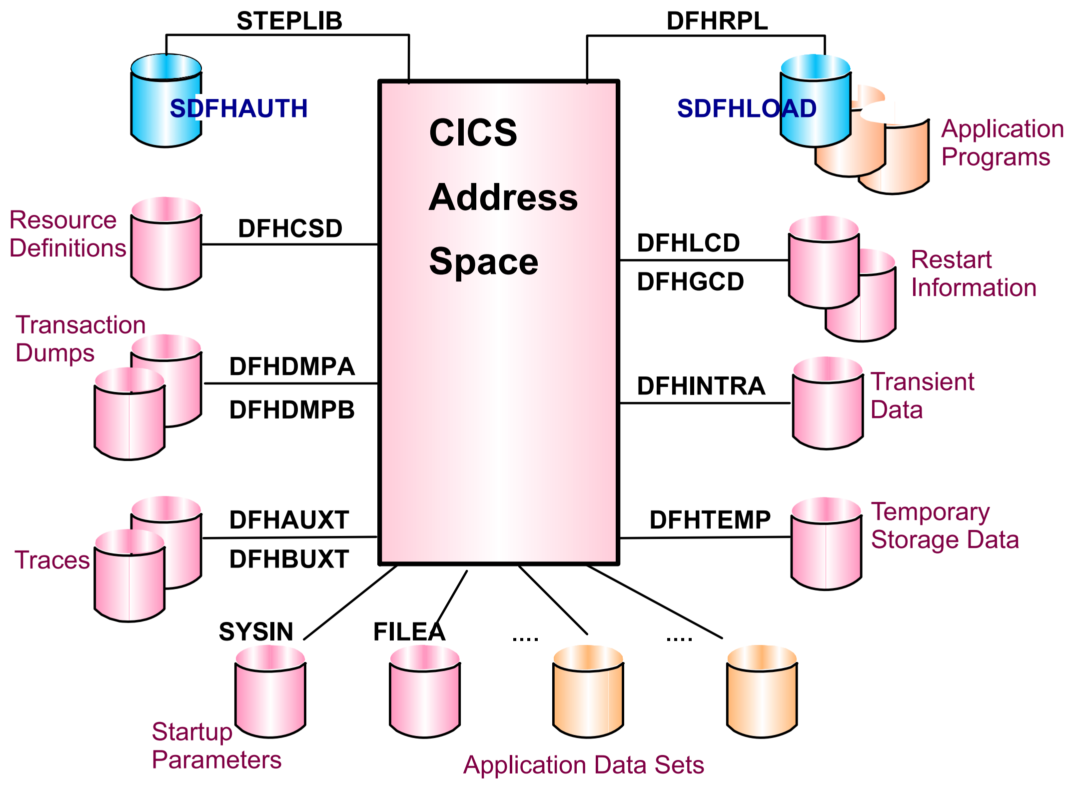


Figure 3-16. CICS TS for z/OS Region Data Sets CI207.0

Notes:

The visual shows the data sets that are required for starting and executing a CICS region which supports the traditional type of CICS applications (for example, non-Java).

The names shown in the inner circle are the ones that appear as DDNAMEs in the CICS startup JCL. The outer explanation says what kind of data is stored there.

- The installation-supplied product libraries, SDFHAUTH and SDFHLOAD, are accessed through the STEPLIB and the DFHRPL ddnames, respectively.
- Application program code is normally provided in separate load libraries that have to be concatenated to the DFHRPL dd statement.
- User files that are under the direct control of CICS TS for z/OS may be defined by DD statements in the startup JCL (details will be discussed in the file control unit); the FILEA shown here is a sample application file used for installation verification, and is considered to be part of the region setup.
- From the other data sets shown, all but the DFHCSD file, which contains resource definitions, and the system parameter SYSIN library, are used exclusively by a single CICS region, and have to be created accordingly per region.

- page intentionally left blank -

DFHISTAR Post-Installation Support

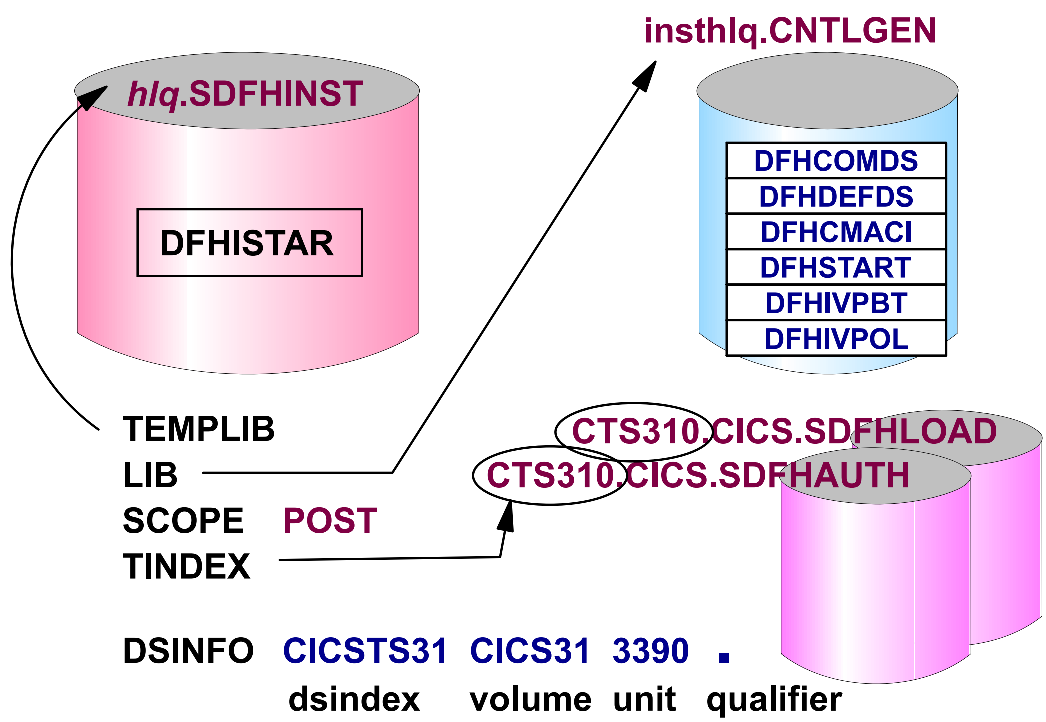


Figure 3-17. DFHISTAR Post-Installation Support CI207.0

Notes:

Running the DFHISTAR job generator with the **SCOPE POST** option generates the post-installation jobs that you can use to create the CICS TS data sets and run the IVPs. This function is available no matter which mode of base installation you used.

- You specify your target SDFHINST installation library as the source libraries for job skeletons.
- The **LIB** option specifies the name of the output library, where the generated JCL is put. This library will be created if it does not yet exist.
- With the **TINDEX** parameter, you specify the high-level index for the CICS TS target libraries, which is used to name the occurrences of the product load libraries shown in the JCL generated.
- With the **DSINFO** parameter, you specify the CICS system data sets that will be created by the jobs **DFHCOMDS** and **DFHDEFDS**:
  - **dsindex** assigns a common high-level index to all system data sets. It may be a one- or two-level name.

- **volume** is the volume identifier (volser) for all the system data sets.
- **unit** is the UNIT parameter for the volume.
- **qualifier** is a partial qualifier added to the index for the data sets created by DFHDEFDS. It may be up to four digits (alphanumeric) in length and, if specified, it is added to the characters "CICS" to make the qualifier. If just a period (.) is specified, no qualifier is used.

- The **DFHSTART** member generated contains the general JCL to start up the CICS region, using the system data sets generated by the jobs named above.
- **DFHIVPBT** and **DFHIVPOL** contain CICS startup JCL as well, with reference to particular control data that supports the regular batch and online verification, respectively.
- For detailed documentation, refer to *CICS Transaction Server for z/OS V3.1 Program Directory, GI10-2586*, and *CICS Transaction Server for z/OS Version 3.1 Installation Guide, GC34-6426.*

DFHISTAR Generated Region Setup JCL

- **Creating CICS Region Data Sets**
  - **DFHCOMDS** — **DFHCSD, SYSIN** — Data sets shareable between CICS regions
  - **DFHDEFDS** — **DFHGCD, DFHLCD, DFHTEMP, DFHINTRA, DFHAUXT, DFHBUXT, ...** — Data sets unique to each CICS region
  - **DFHCMACI** — **DFHCMACD** — CICS Messages & Codes online
- **Setup Region Start-Up JCL (procedure)**
  - **DFHSTART** — Determine execution - run CICS format dumps and traces

Figure 3-18. DFHISTAR Generated Region Setup JCL CI207.0

Notes:

The jobs generated by DFHISTAR for setting up a CICS region environment are grouped as follows:

- **DFHCOMDS** is used to create the data sets that may be shared between multiple CICS regions. Mainly, these are the DFHCSD resource definition file and the SYSIN library containing the startup parameter entries for particular processing, such as the installation-provided IVP.
- **DFHDEFDS** creates all the system data sets that are needed by a CICS region exclusively. You may review the data sets' SPACE request specifications and tailor them to your needs, but you should not change any DCB or VSAM attributes.
- The **DFHSTART** member produced contains a procedure made up of seven steps:
  - In the first two steps, where no program is run, return codes are set that determine whether or not CICS is really executed (third step), and whether or not the CICS dump and trace data is formatted (fourth to seventh step).
  - Step 3 is the real execution of CICS (`EXEC PGM=DFHSIP`).

- Steps 4 and 5 format dump data contained in the region's dump data sets.
- Steps 6 and 7 format trace data contained in the region's trace data sets.

The DFHSTART procedure may be activated as-is in a procedure library, and invoked as a started task by a z/OS START command. If you run CICS as a batch job, you call the procedure with an EXEC statement in your JCL. In both cases, parameters have to be provided that name the index parts of the system data sets and the YES/NO decision about running CICS, or the dump and trace formatting steps, or both.

Most installations tend to isolate the step which executes CICS TS and run the dump and trace formatting separately.

Two more remarks as to the JCL generated:

- As the **DFHCMACD** messages and codes file is referenced by a DD statement within the DFHSTART procedure, the **DFHCMACI** job has to be run in order to create this file and load the contents from the installation SDFHMSGS library*.*
- In addition to the data sets shown in the last but one visual, the following data sets are created by DFHDEFDS and referenced by the DD names in the DFHSTART procedure:

DFHLRQ	Local Request Queue data set: used by the Business Transaction Services (BTS) feature
DFHEJOS	Object Store data set, used for EJB support
DFHEJDIR	EJB Directory data set, used for EJB support
DFHAFDEM	EJB Event data set, used for EJB support
DFHBRNSF	Bridge Link 3270 data set, supporting the enhanced 3270 bridge facility in CICS TS 2.2
DFHHTML	Web Template data set, supporting the CICS Web Support function
BANKACCT	Sample application data set for IIOP server support
DFHDPFMB	Debugging profiles base data set
DFHDPFMP	Debugging profiles path data set

If you are sure that one or more of the named functions will not be used by a particular region, you may remove the appropriate entries from the JCL when running the DFHDEFDS job and from the CICS startup JCL.

Of course, the data sets may be created at some later time and appropriate entries added to the startup JCL, when needed.

Installation Verification Procedures

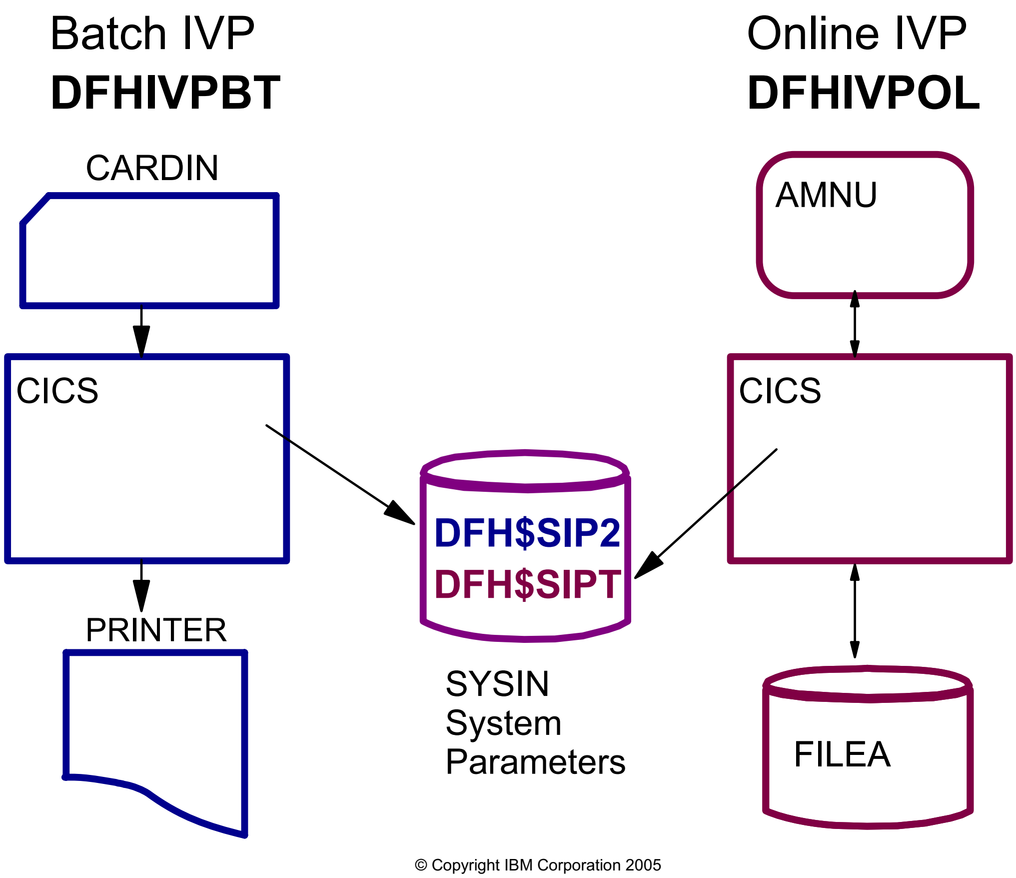


Figure 3-19. Installation Verification Procedures CI207.0

Notes:

CICS supplies installation verification procedures (IVPs) to confirm that CICS is operational: one that does some batch command processing only, and another that requires a terminal user to log on to CICS and run a CICS-supplied sample application. You are advised to run these IVPs before you implement your own applications.

- Batch and Online IVP is provided as two jobs (members) generated by DFHISTAR with the names shown in the visual.
- The two members mainly differ in the initialization parameter input that is referenced by the SYSIN DD statement, thus causing a different behavior of the CICS region. Remember, the SYSIN data set is created by job DFHCOMDS as part of the region setup, and primarily is used to support the IVPs.
- The **Batch IVP** uses a CICS facility called *sequential terminals* as its input/output device. Data stored in a sequential data set is accessed by CICS after successful initialization, and processed as if it were entered from a terminal. Any output produced by the transactions invoked are written to another data set. Both input and output data

sets have to be referenced in the CICS startup JCL. More information about *sequential terminal devices* is provided in a later unit.

The sample input provided by the installation ends with the command to shut down CICS. Thus, if everything runs fine, the DFHIVPT batch IVP job terminates normally, typically after some seconds of processing.

- The **Online IVP** requires an active CICS VTAM definition, and a terminal that is capable of logging on to this VTAM applid. Successful logon without any further configuration depends on the compatibility of some VTAM definitions with CICS-supplied terminal specifications. Assuming that the logon succeeds, further IVP processing is based on a CICS-supplied sample application, which is invoked by transaction code AMNU and provides functions to read to, and write from, the sample application file named FILEA, which was created and loaded by the DFHDEFDS post-installation job.
  The region started with the DFHIVPOL online IVP job, and will not terminate unless the shutdown command is entered by hand from a terminal (or a console).

  The DFHIVPOL JCL references SYSIN member DFH$SIP2 to provide certain startup parameters, one of which is the VTAM applid to be used. This is specified as CICSIVP, which is likely to be replaced by a valid VTAM applid.
- There is a third CICS-supplied IVP job, named **DFHIVPDB**, which ensures that DL/I (IMS, DBCTL) databases are accessible from within CICS. This requires some additional configuration, both on the CICS and the DBCTL (IMS) side.
- All IVP processes, their input and output, and the objects used, are documented in detail in the *CICS Transaction Server for z/OS Version 3.1 Installation Guide, GC34-6426*.

CICS Construction at Startup

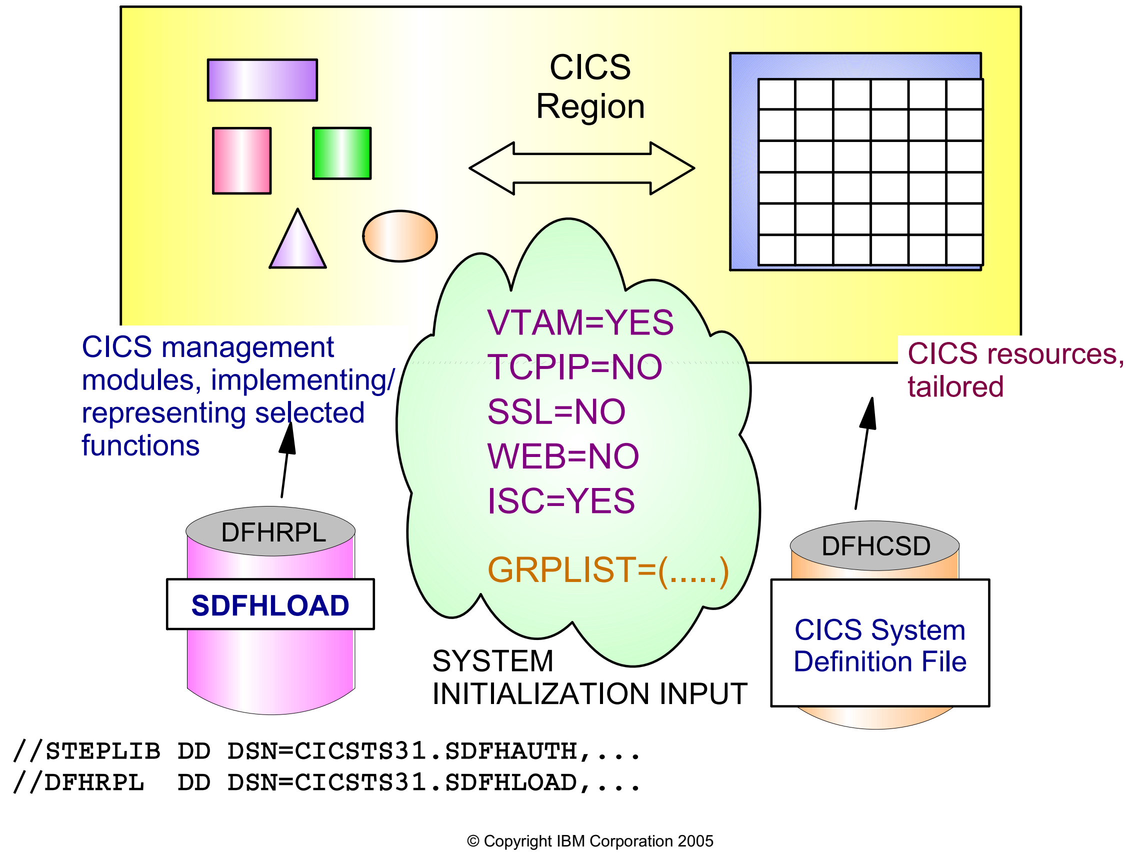


Figure 3-20. CICS Construction at Startup CI207.0

Notes:

- When CICS TS is brought up, programs and data have to be loaded from specific sources. As with other products running in a z/OS environment, the search paths for any modules to be accessed are determined by DD statements within the startup JCL, unless the modules are contained in an MVS linklist library.
- CICS management modules, programs, resources, and controls defined with macro tables are loaded from the **DFHRPL** (Relocatable Program Library) DD concatenation. CICS management modules reside in the installation SDFHLOAD library.
  - A number of functions, which are not mandatory for CICS execution, can be selected by means of system initialization options. In most cases, the specification appears in the form FUNCTION=YES or NO.
- Resources defined by RDO (Resource Definition Online) are loaded from the file identified by the **DFHCSD** DD statement. Which subset of resources defined in the DFHCSD file is actually activated for use is determined by the system initialization parameter **GRPLIST=**; details are provided in the next unit.

Providing CICS Initialization Parameters

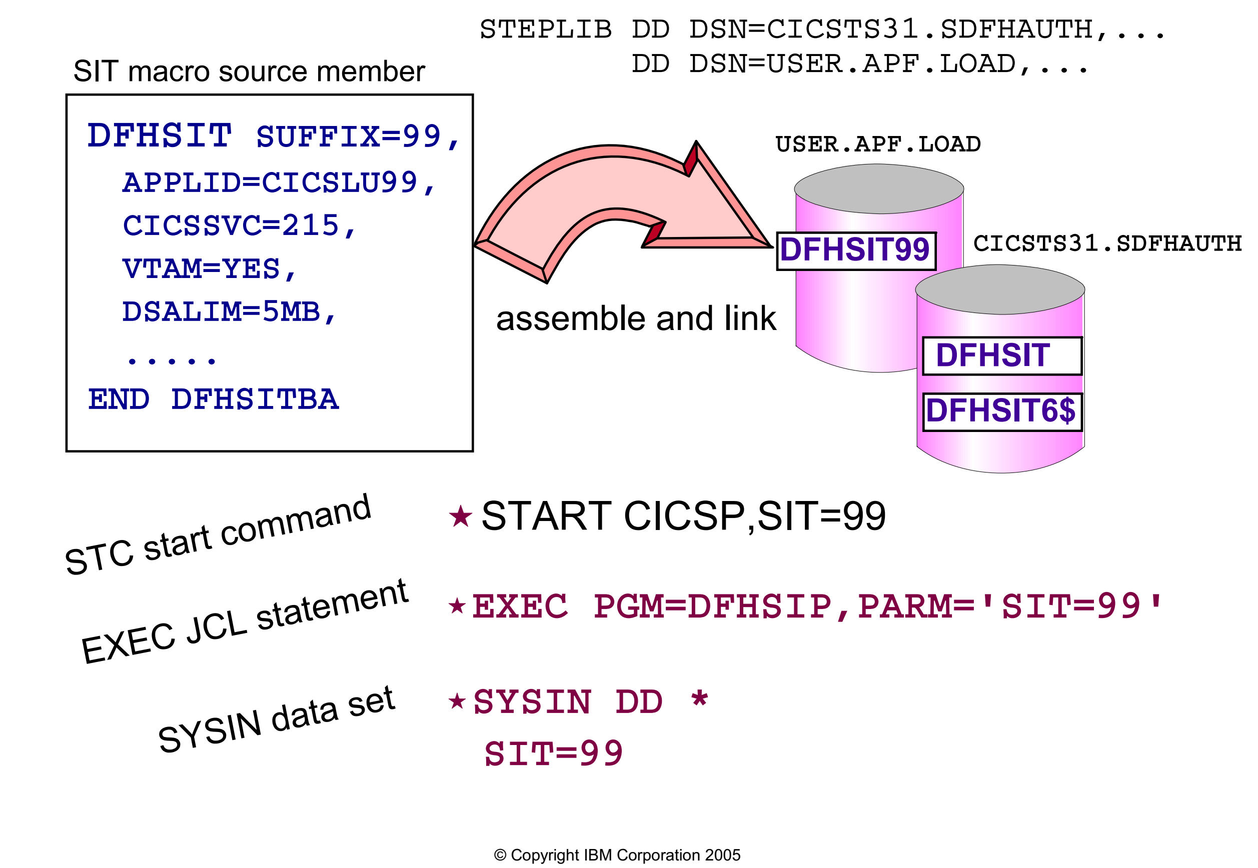


Figure 3-21. Providing CICS Initialization Parameters CI207.0

Notes:

- The primary way to provide the variable control information needed to determine the system initialization processing is the **System Initialization Table (SIT)**.
- The SIT is built with a macro named **DFHSIT,** which is supplied in the CICS macro library SDFHMAC.
  The resulting load module is named DFHSITxx, where “xx” is a two-digit alphanumeric suffix, which itself is specified as a parameter within the macro. It must be placed in a library concatenated to the product SDFHAUTH library, within the STEPLIB DD concatenation.
- The DFHSIT macro contains about 150 single parameters addressing all the variable options for CICS initialization and processing, a lot of which will be discussed throughout the course.
- On CICS startup, the SIT module to become active is selected with the **SIT=xx** parameter provided in one of three possible ways:
  - As a startup-parameter specified with the started task START command

  - Within the PARM string of the JCL EXEC statement
  - Within the SYSIN data set (or instream, as shown) referenced by the startup JCL.

- If no SIT= parameter is specified, the unsuffixed default SIT (the member named DFHSIT) is loaded from SDFHAUTH.
  Another installation-provided parameter collection is **DFHSIT6$**, which is a good basis to start for many environments. This is the SIT table we will use for our lab exercises.

- Apart from determining the suffix of the SIT load module, many real SIT parameters may be specified by the three methods shown above. These parameters complete the ones contained in the SIT load module, and take precedence over them, if the same arguments have already been used in the SIT source. This technique, called “**SIT override**, is covered in more detail in a later unit.

More CICS TS Macro Tables

Table Name	Macro	Usage and Remarks
Command List Table	DFHCLT	Commands controlling an XRF takeover
Data Conversion Table	DFHCNV	To convert data structures for communication, such as with ASCII systems
Monitor Control Table	DFHMCT	To control user event monitoring
Program List Table	DFHPLT	Automated processes at CICS startup and shutdown
System Initialization Table	**DFHSIT**	**CICS initialization parameters and options**
System Recovery Table	DFHSRT	Define special recovery options
Terminal List Table	DFHTLT	Grouping terminals
Transaction List Table	DFHXLT	Special handling for named transactions on CICS shutdown
Temporary Storage Table	DFHTST	Special processing for temporary storage
File Control Table	DFHFCT	BDAM files resource definitions
Terminal Control Table	DFHTCT	Resource definition for certain terminal types

Figure 3-22. More CICS TS Macro Tables CI207.0

Notes:

There are a number of optional macro tables that can be used to control certain functions of CICS, as shown in the table above.

- These tables are prepared in the same way as the SIT, with a two-digit suffix specified in their initial entry, except for DFHCNV, which is unsuffixed. The load modules have to be placed into a library which is concatenated in the DFHRPL DD statement of the CICS startup JCL.
- Most of them are activated through SIT parameters by naming the table name and the suffix of the table occurrence to be loaded, for example `MCT=01` or `SRT=AX`. The PLT and XLT table are involved in shutdown processing; they may be specified with the command to terminate CICS.
  If the function covered by a table is not used, this is specified by a `tablename=NO` entry in the SIT, for example `TLT=NO`, which is the default.
- The three tables at the bottom are used for certain resource definition means. They will be covered in the appropriate units later.

Installation and Verification (Summary)

Figure 3-23. Installation and Verification (Summary) CI207.0

Notes:

- Before installation, read the Installation Guide and Program Directory.
- Call the Support Center to obtain the latest information.
- Make sure your MVS is set up properly to support CICS TS.
- Tailor DFHISTAR to meet your requirements.
- Run the jobs created by DFHISTAR.
- Integrate into the operating system.
- Verify your installation.

Unit Summary

Having completed this unit, you should be able to:

- Name the most important hardware and software requirements to run CICS TS
- Describe the flow of steps necessary to install CICS TS
- Add CICS procedures to your procedure library
- Integrate CICS into the operating system
- Create CICS system and test data sets
- Verify CICS installation with supplied procedures

Figure 3-24. Unit Summary CI207.0

Notes:

Unit 4. Resource Definition Online

What This Unit Is About

This unit will familiarize you with the system integrated online application that is supplied to maintain resource definitions within a running CICS.

It teaches you the concepts of Resource Definition Online (RDO), and how to define and manage resources, groups, and lists.

What You Should Be Able to Do

After completing this unit, you should be able to:

- Identify resources that can be defined online
- Explain the relationship between resources, GROUPs, and LISTs
- Manage the resources in GROUPs and LISTs
- Identify which resource definitions CICS loads at initialization
- Dynamically install resource definitions while CICS is active
- Manage the RDO data set

How You Will Check Your Progress

- Machine exercise lab.
  You will create your own CSD file and bring in the resource definitions necessary to start up your CICS region using the RDO batch utility.

References

CICS System Definition Guide, SC34-6428

CICS Resource Definition Guide, SC34-6430

CICS Operations and Utilities Guide, SC34-6431

Unit Objectives

After completing this unit, you should be able to:

- Identify resources that can be defined online
- Explain the relationship between resources, GROUPs, and LISTs
- Manage the resources in GROUPs and LISTs
- Identify which resource definitions CICS loads at initialization
- Dynamically install resource definitions while CICS is active
- Manage the RDO data set

Figure 4-1. Unit Objectives CI207.0

Notes:

4.1 RDO Organization

Offline Preparation of Macro Tables

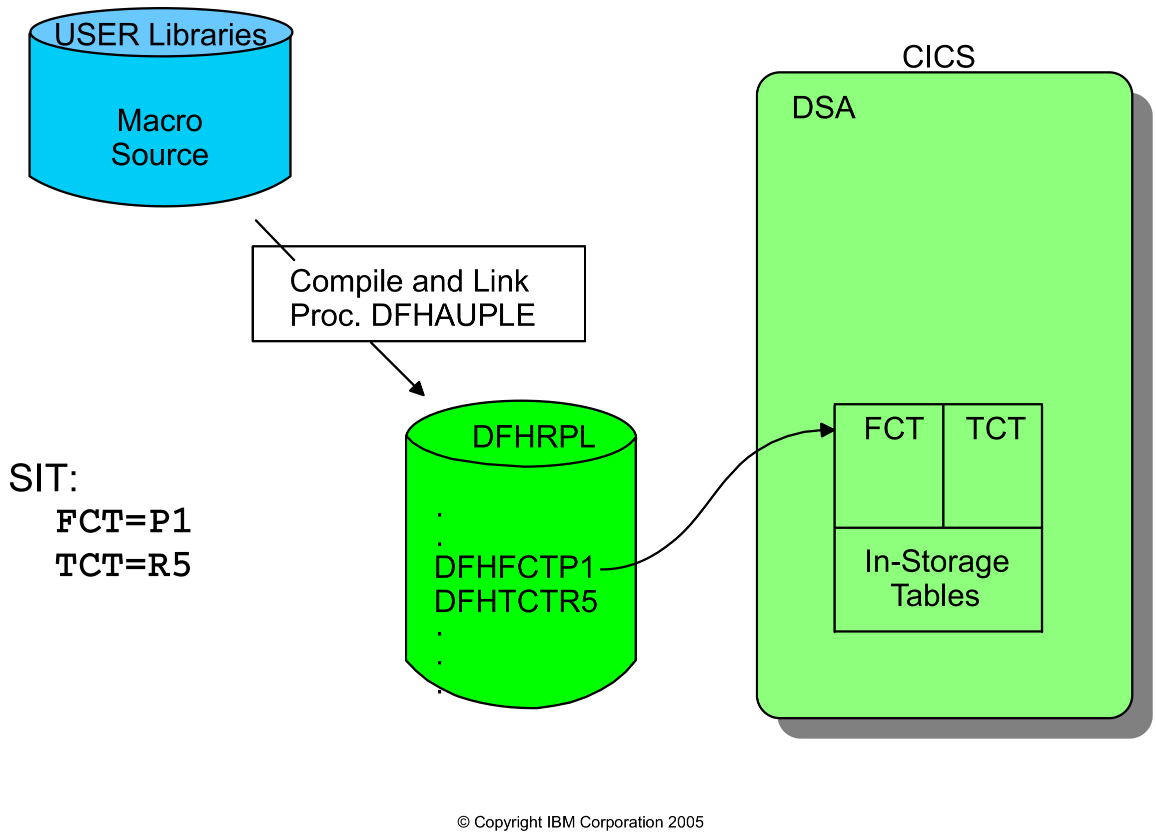


Figure 4-2. Offline Preparation of Macro Tables CI207.0

Notes:

Using macro tables has, in the past, been a method of defining resources to CICS. In the meantime, this method has almost disappeared. Only some special file and terminal facilities require a resource definition macro table.

The principle of handling these tables is/was as follows:

- You assemble and link CICS macro-generated tables into a load library. When CICS starts, it loads these resource definitions into the in-storage tables.
- To make changes, you edit the tables, reassemble and link them, then recycle CICS to incorporate the changed definitions.

Online Method

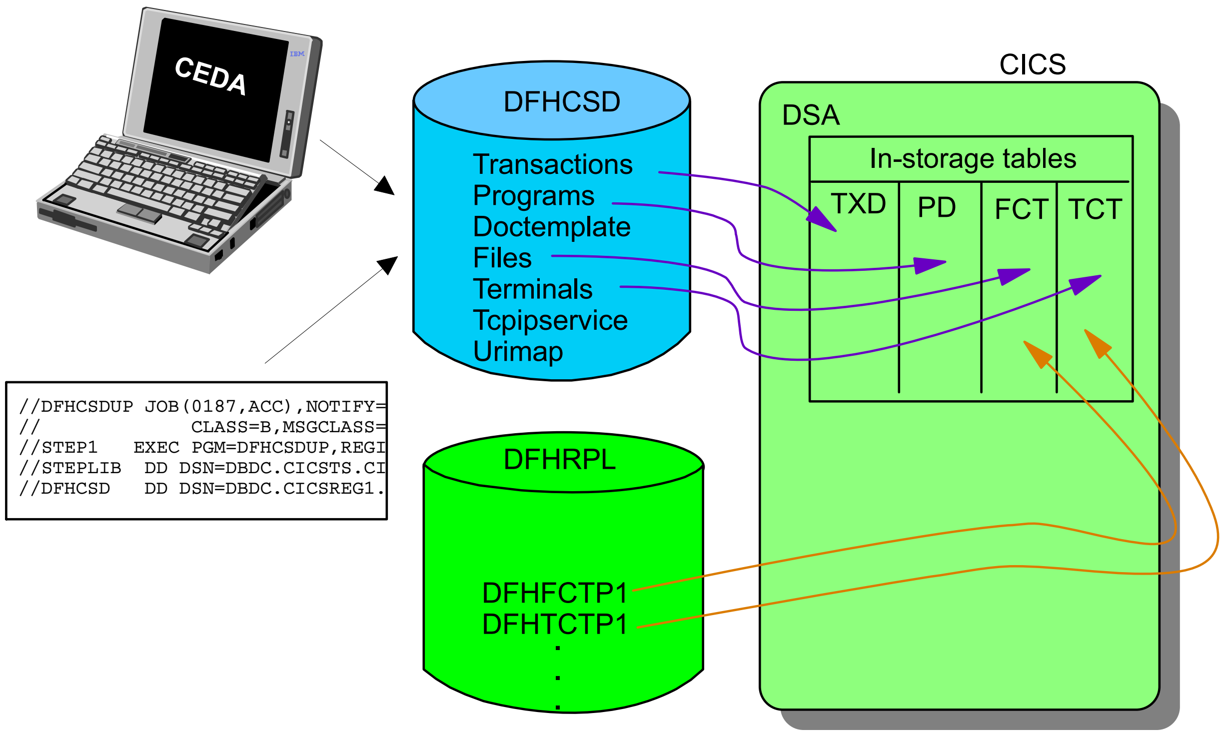


Figure 4-3. Online Method CI207.0

Notes:

- The regular and more comfortable method is to define resources in the CICS System Definition (CSD) file using a system-provided transaction.
- CICS loads resource definitions from the CSD file into the in-storage tables at initialization, as with macro tables, except that the definitions are built dynamically in the DSA as part of the (cold) start-up process.
- This allows you to add or change resources without terminating CICS.

Advantages:

- Increase availability
- Reduce scheduled CICS outages
- Ease of use

Resources Defined with RDO

	RDO	Macro
Programs	X	———
Map sets	X	———
Transactions	X	———
Terminals	X	TCT
Files	X	FCT
Doctemplates	X	———
Web Services	X	———
Temporary Storage Queues	X	TST
Journal Models	X	———
TCP/IP Services	X	———

Figure 4-4. Resources Defined with RDO CI207.0

Notes:

New resource types are defined with RDO. This chart shows the most important resource types that you can define with RDO, and the equivalent macro definition, if appropriate.

- VTAM terminals can be defined only with RDO, while sequential terminal definitions (card reader/line printer) are supported only with macro.
- Similarly with files: you must define VSAM files with RDO, but you must create a macro File Control Table (FCT) for defining BDAM files.
- For VTAM terminals and VSAM files, you can use macro definitions only to assemble a table for migration to RDO. The same is true for Transient Data Queues and the DCT macro table. The process of migration will be covered later.
- PPT and PCT cannot be assembled with CICS TS for z/OS.

RDO Organization: Single Resources

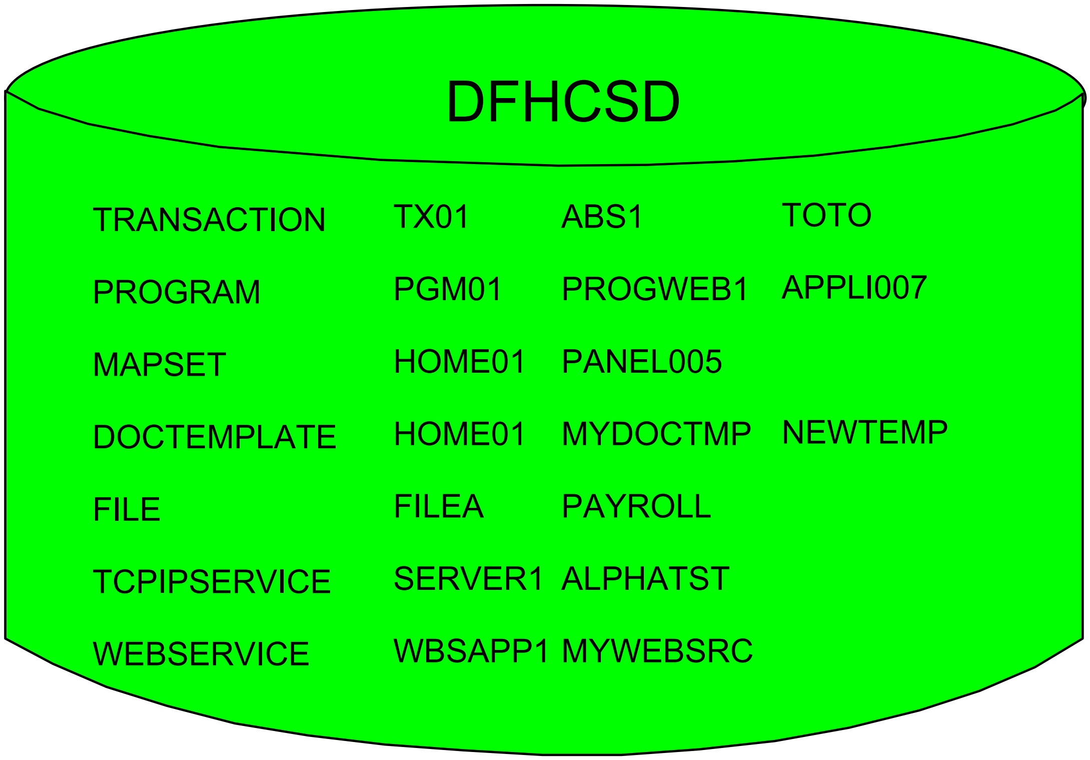


Figure 4-5. RDO Organization: Single Resources CI207.0

Notes:

- Single resources are identified by:
  - Their resource type
  - Their name, which may be of different length, depending on the type of resource, for example:
    - Terminal names ("termids"), connection names, transaction codes, and transient data queue names have four digits in length
    - Program and map set names and most of the other resource types may have names with up to eight characters in length
    - Temporary storage queue names may be 16 digits in length, if defined via RDO.
- Resources of different types may have equal names.

RDO Organization: Groups

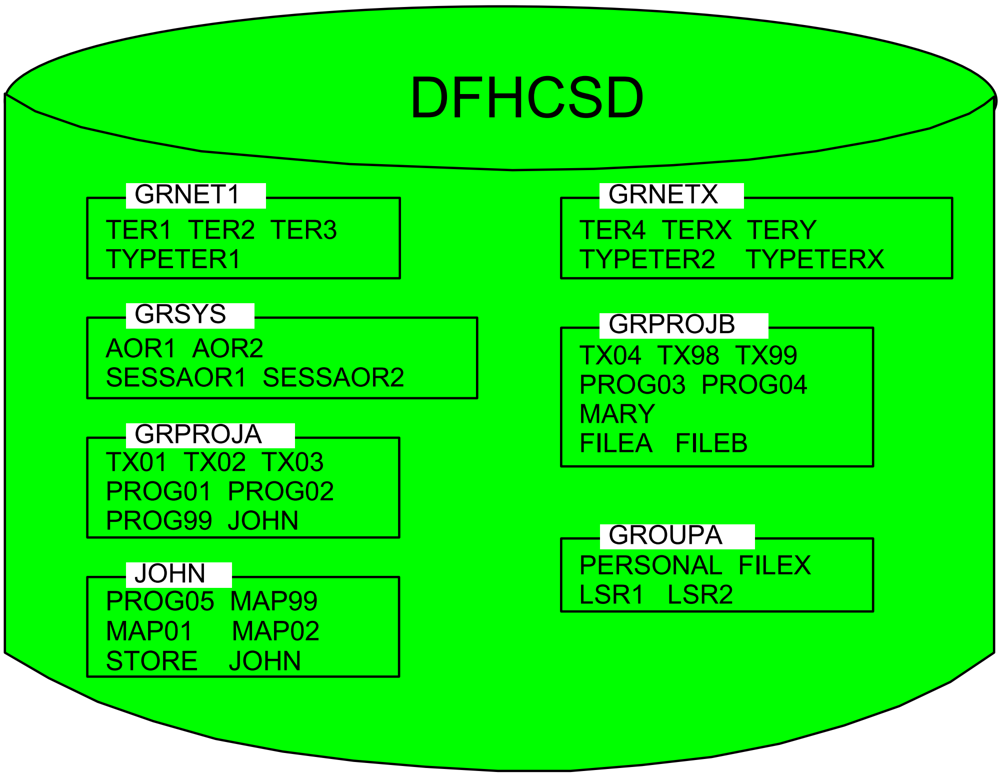


Figure 4-6. RDO Organization: Groups CI207.0

Notes:

- Each resource is part of a group;
  this is enforced by RDO syntax, which is as follows:

  `(CEDA)` **DEFINE restype(resname) GROUP(grname)**

  for example:

  `(CEDA)` **DEFINE PROGRAM(BALANCE) GROUP(CREDIT)**

- The first resource defined in a group causes the group entry to be created.
- Group names are arbitrary, maximum 8 characters in length.
- When placed into separate groups, there may be equally named resources of the same resource type.
- Resource types may be mixed within a group.
- Keep group sizes in a range easy to handle online.

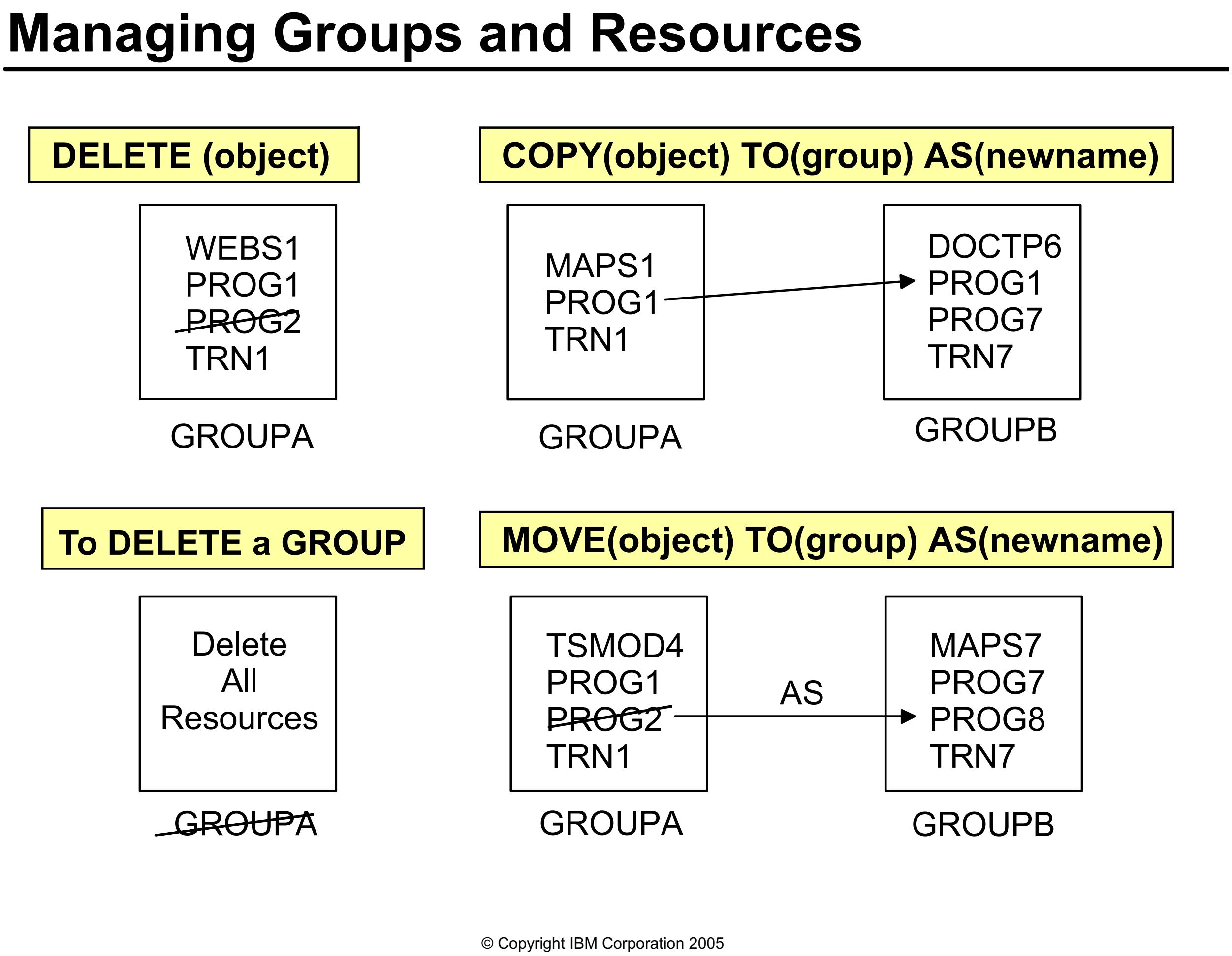


Figure 4-7. Managing Groups and Resources CI207.0

Notes:

- RDO commands allow you to easily manage groups and resources in the CSD file.
- When you DELETE all resources from a group, the group is also deleted.
- You can COPY and MOVE resources from one group to another. The MERGE and REPLACE options tell RDO what to do in case the TO group already contains the resource.
- Use the **AS** keyword to rename a resource as you copy or move it:

```
MOVE G(GROUPA) PROG(PROG2) TO(GROUPB) AS(PROG8)
```

IBM-Supplied Definitions

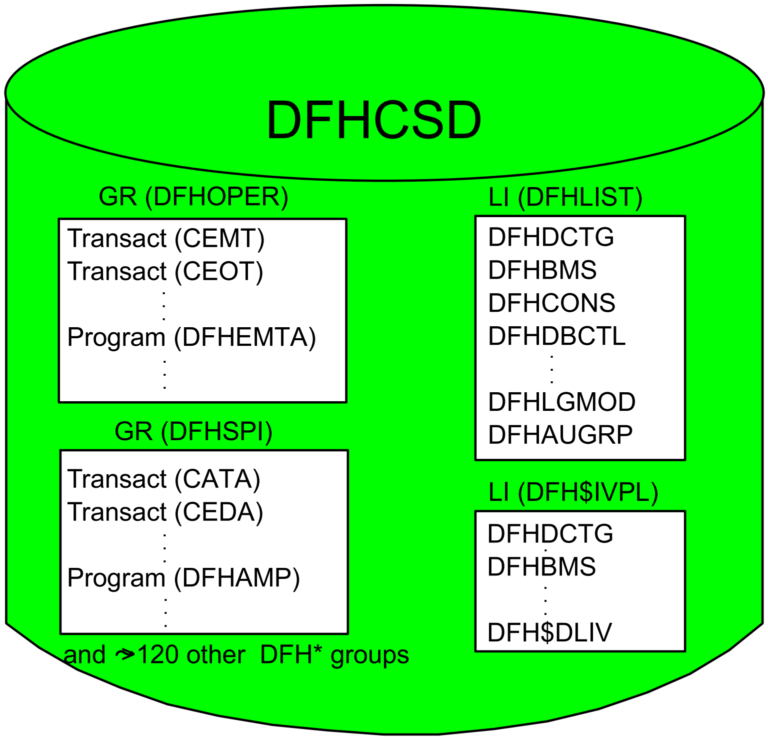


Figure 4-8. IBM-Supplied Definitions CI207.0

Notes:

- CICS TS uses the same mechanisms that are offered to applications for its internal system functions:
  - Internal processes are run by transactions and programs.
  - Some CICS-supplied transactions provide a user interface comprised of a number of map sets.
  - As a whole, more than 1,000 single resource definitions are provided.
- These resources have to be defined just as application resources.
- These definitions are supplied by the product, namely with SDFHLOAD(DFHCURDI)
- By means of a utility function (to be covered later), these definitions are placed in your CSD file.
- All groups are named DFH* .
- These IBM-supplied resource definitions are protected, in other words RDO does not allow them to be changed.

RDO Organization: Lists

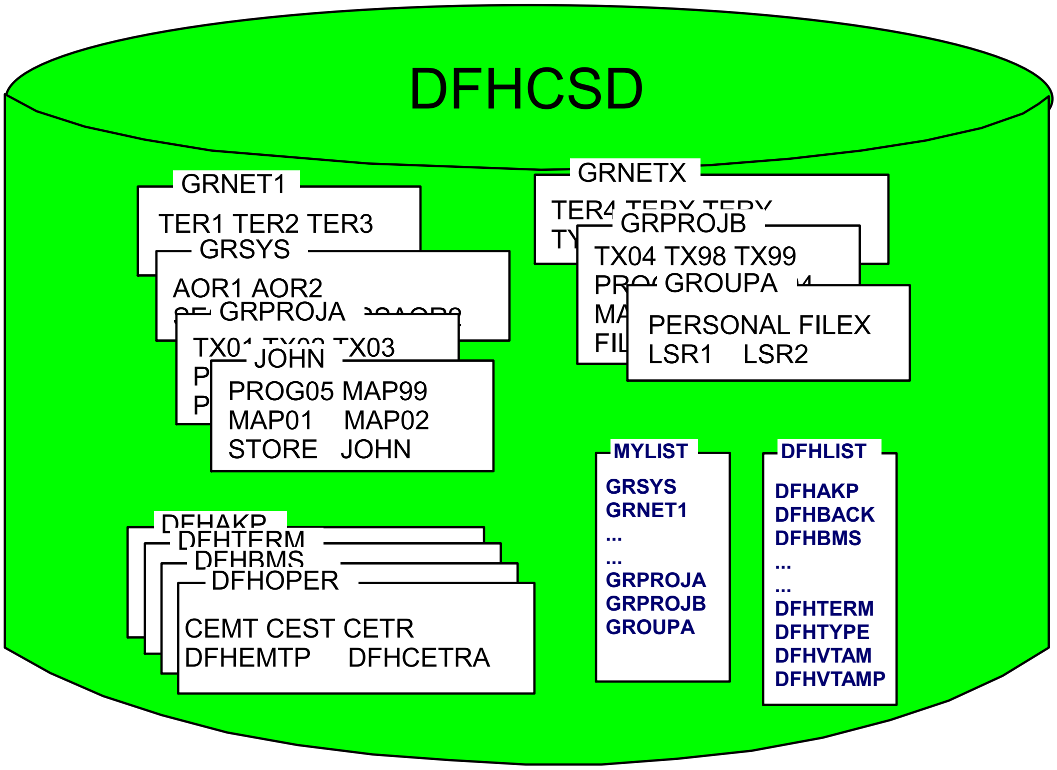


Figure 4-9. RDO Organization: Lists CI207.0

Notes:

- Lists are collections of group names.
- Lists are used to determine what resources, based on RDO groups, are to be activated for a particular CICS execution.
- Lists will only be honored during initialization, when CICS performs a COLD or INITIAL start (these terms will be explained in detail later).
- IBM-supplied lists DFHLIST and DFH$IVPL contain the group entries for those resource definitions that are required for pure system execution and for the CICS-supplied installation verification process, as documented in the *CICS Transaction Server for z/OS Version 3.1 Installation Guide, GC34-6426*.

GRPLIST Processing

GRPLIST=(DFHLIST,LIST1,MYLIST*,SAMPLIST)

- Up to 4 list names, generic allowed
- One list for multiple groups

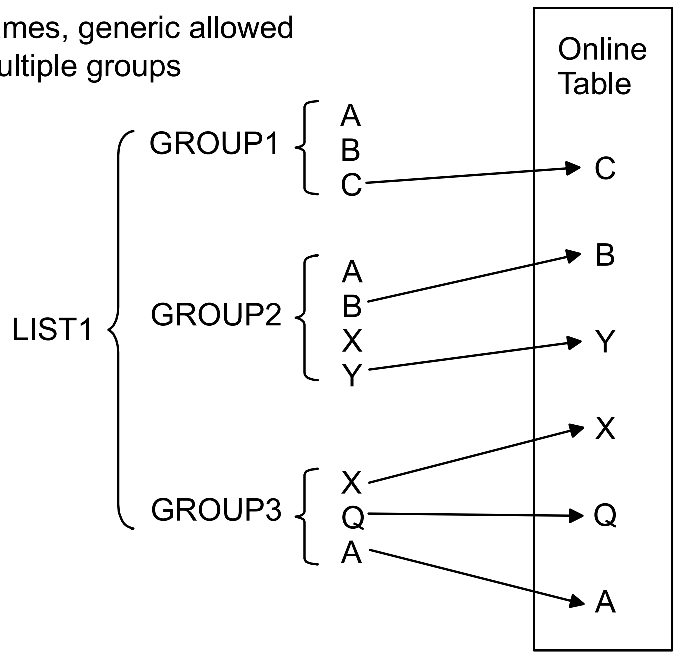


Figure 4-10. GRPLIST Processing CI207.0

Notes:

- Up to four list names, including generic names, may be specified with the SIT parameter GRPLIST=.
- The groups are installed in order of appearance in the list.
  - So, in the event of duplicate resource names, the last duplicate wins, *if the appropriate resource can be over-installed*.
  - Over-installing of existing resources will be performed on certain prerequisites only, and for transient data queues and DB2 related resource definitions, over-installing is not supported at all at startup time.
- The reason for allowing duplicates is to allow you to test modifications easily.

In the example above, which version of resource 'A' will be used, if we assume that 'A' is a resource that allows for over-install?

Managing Lists

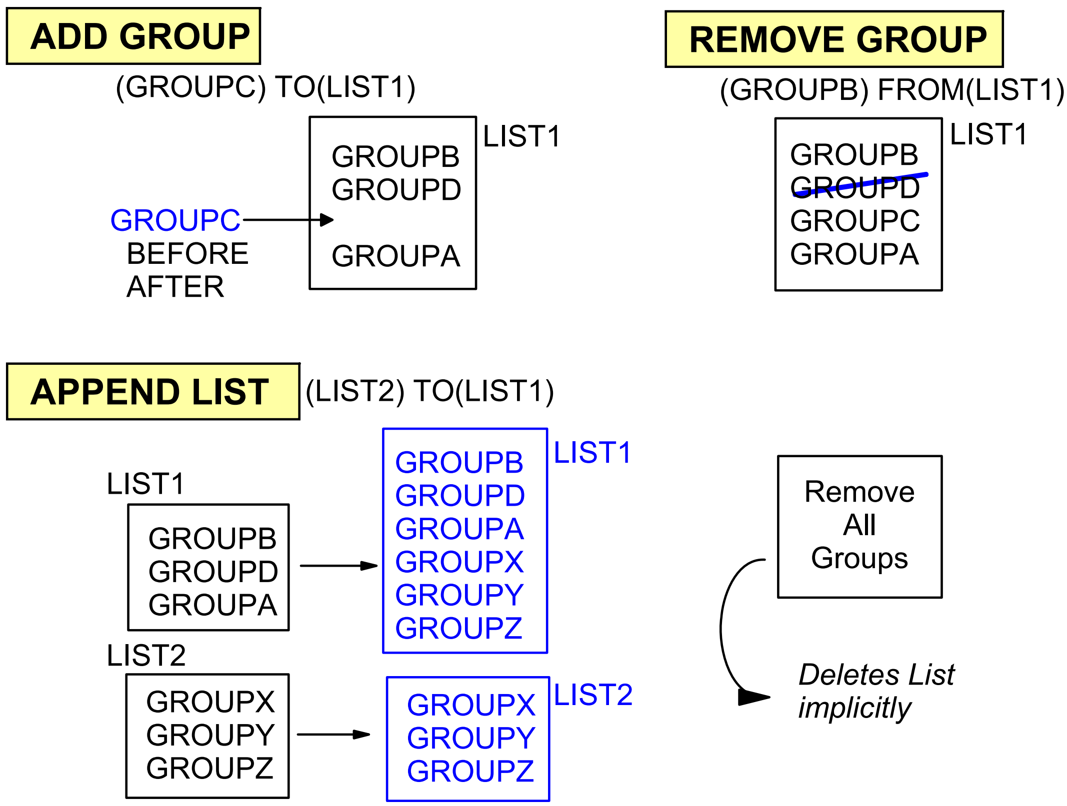


Figure 4-11. Managing Lists CI207.0

Notes:

- A list is created implicitly by one of the following functions:

  `ADD GR(grname) LI(listname)` - the new list contains a single group entry.
  `APPEND LI(listnam1) TO(listnam2)` - the new list is a copy of the existing one.

- When you add groups to an existing list, you can control the resulting sequence by specifying either BEFORE or AFTER on the ADD command. The default position is at the end of the list.

- You can REMOVE a group from a list. This does not delete the group or its resources, but removes its association with the list.
  `REMove Group(GROUPD) List(LIST1)`

- **When you remove the last group from a list, CICS deletes the list.**

- One list may be APPENDed to another existing list.
  `APpend List(LIST2) To(LIST1)`

- A group may be referenced by more than one list, or by no list at all, if the appropriate definitions are not yet or no longer used.

4.2 Installing Resources

Installing Resource Definitions

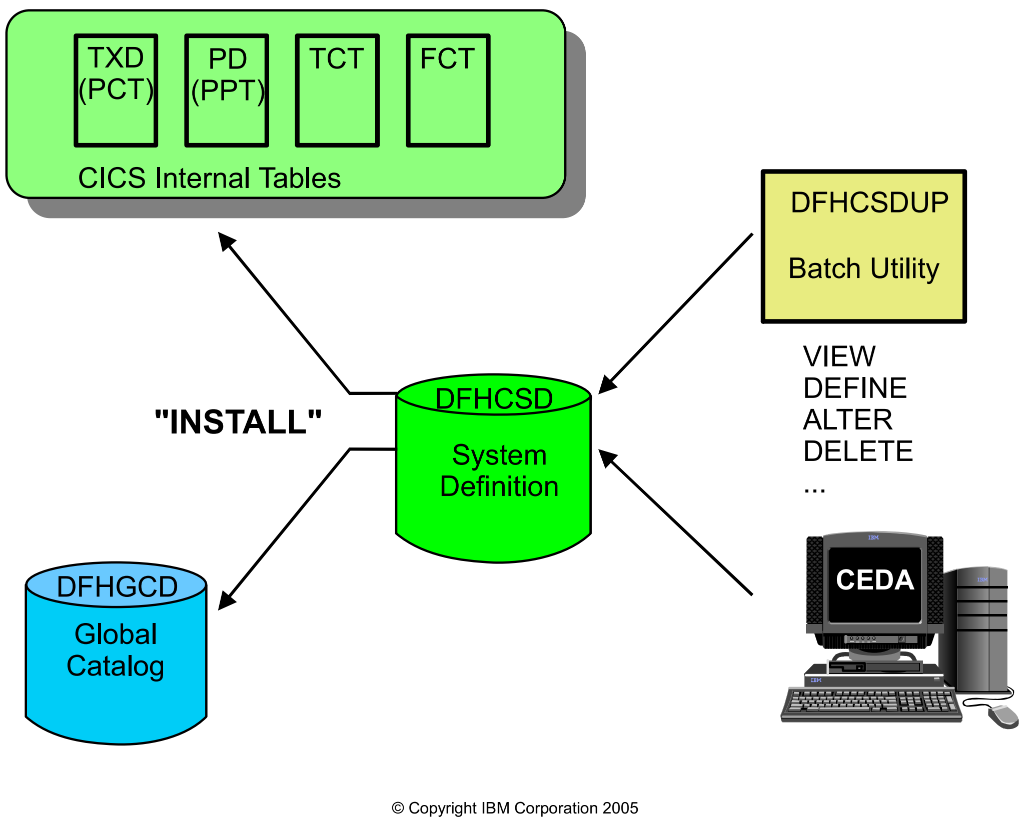


Figure 4-12. Installing Resource Definitions CI207.0

Notes:

- INSTALL is the process of activating defined resources by loading their definitions into the in-storage tables.
- For restart purposes, definitions are stored in the Global catalog data set at install time.
- At CICS initial start or cold start, all resource definitions that are determined via the SIT GRPLIST parameter are INSTALLED.

  The CICS message log will contain a message for every single resource that has been installed during startup.

Dynamically Installing Resources

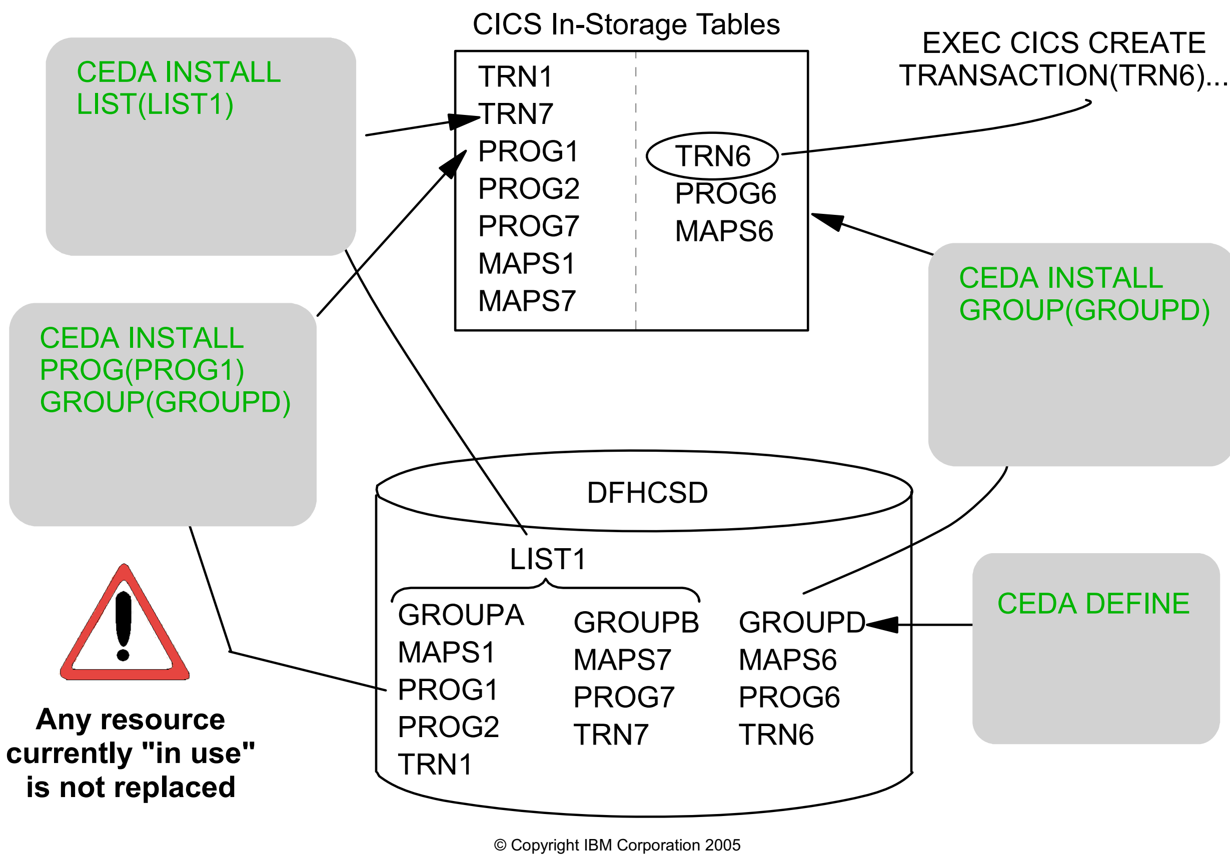


Figure 4-13. Dynamically Installing Resources CI207.0

Notes:

- You may invoke dynamic installation of resources at the level of single resources, groups, or lists; but actual installation always occurs on the resource level.
- The `INSTALL` command can be entered from a `DISPLAY` or an `EXPAND` result screen as a line command to the appropriate object.
- Whether or not over-installing of existing resources succeeds is determined per single resource by the same rules, no matter what range of resources you install at a time:
  - Any resource currently “in use” is not replaced.
  - In the event of group or list-level install, the process continues with the next resource.
  - RDO (CEDA) displays messages (use PF9 to retrieve) telling you that the install process has completed only partially, and informs you which resources are in use.
- A way to ensure successful installation of resources is to disable them before attempting reinstallation.
  In case of terminals, “disabling” means setting “out of service”.

Dynamically Discarding Resources

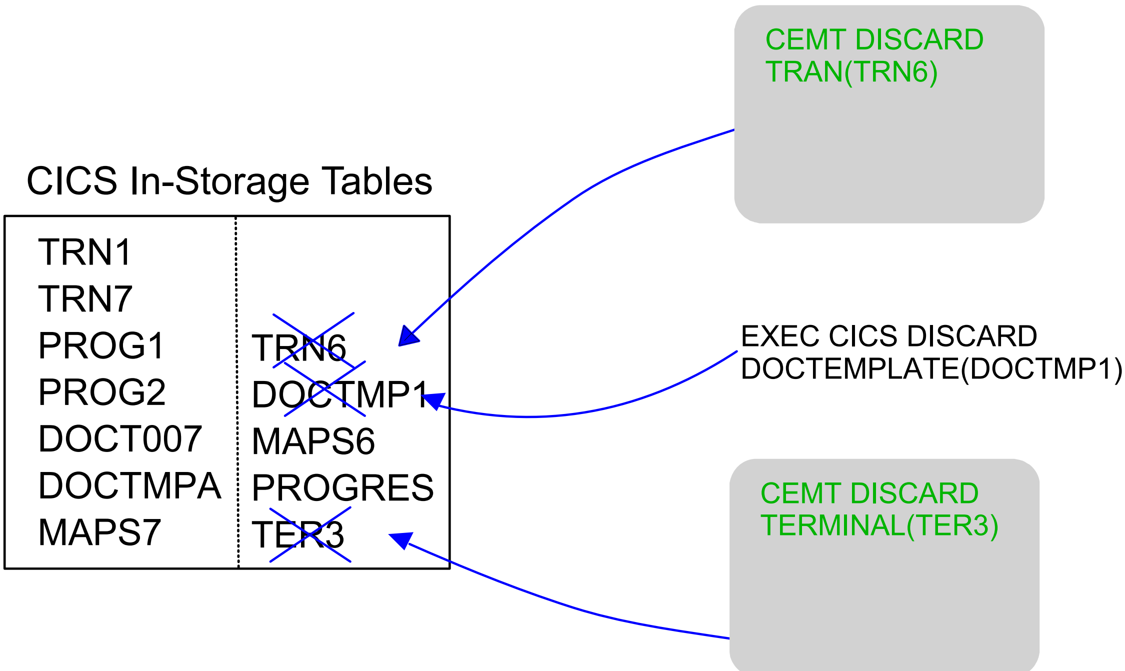


Figure 4-14. Dynamically Discarding Resources CI207.0

Notes:

- With the CEMT **DISCARD** function, single named objects can be removed from the in-storage tables, with the effect that they are no longer active. This option may be thought of as a kind of *uninstalling*.
  - Generic resource names are not supported with the DISCARD option.
- An elegant and safe way to discard resources is to perform a `CEMT INQUIRE` command and then, in the inquire result screen, place a “D” in front of the resource(s) to be discarded.
- *CICS Supplied Transactions, SC34-6432*, describes the CEMT transaction options in detail.
- *Note:* As the CICS runtime environment is “not aware of” RDO groups and lists, these cannot be referenced by `CEMT DISCARD`, nor by other CEMT commands.

4.3 RDO Online Transactions

CEDA/B/C Transactions

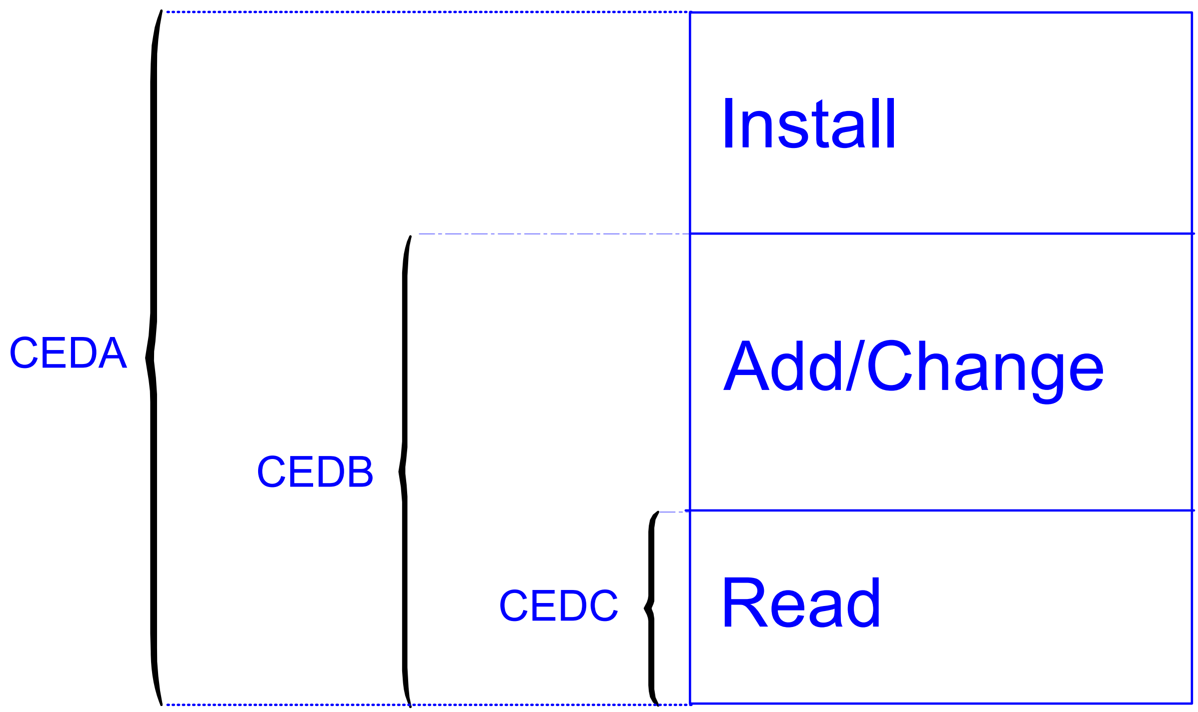


Figure 4-15. CEDA/B/C Transactions CI207.0

Notes:

RDO provides three transactions to support different resource management functions.

The *CICS Resource Definition Guide, SC34-6430*, provides a good tutorial on the use of CEDA/B/C transactions.

Why is it helpful to have three different transactions?

DISPLAY Groups and Lists

CEDA DI G(DFHS*)

GROUP
DFHSDAP
DFHSIGN
DFHSO
DFHSPI
DFHSTAND

CEDA DI LI(*)

LIST
DFHLIST
DFH$IVPL
USRLIST1
USRLIST2

Figure 4-16. DISPLAY Groups and Lists CI207.0

Notes:

- Suppose you want to modify resources but are unsure of the group name. You can DISPLAY using a generic name, to see a display of the groups matching your search argument.
- On the resulting screen, you can:
  - Enter **e** (for expand) after the proper group name, then
  - Enter commands for the resources from the expand screen.
- This combination of a generic display, followed by expand and then other commands, is usually the easiest way to maintain resources if the precise group or list name is unknown.

CEDA EXPAND Subfunction

CEDA EXPAND GROUP(GROUPA)

NAME	TYPE	GROUP	
MAPS1	MAPSET	GROUPA	`copy to(GROUPB)`
PROG1	PROGRAM	GROUPA	`=`
PROG2	PROGRAM	GROUPA	`alter`
TRN1	TRANSACTION	GROUPA	`view`

CEDA E L(LIST1)

NAME	TYPE	LIST	
GROUPA	GROUP	LIST1	
GROUPB	GROUP	LIST1	`rem`
PROJAB	GROUP	LIST1	`add g(GRXYZ)`

Figure 4-17. CEDA EXPAND Subfunction CI207.0

Notes:

- The EXPAND command shows the resources in a group, or the groups in a list.
- After expanding a group or list, you can enter commands — such as alter, copy, remove, and add — to the right of the resource or group name. This is much more convenient than using the command line at the top of the screen.
- See the "Using the RDO transactions" chapter of the *Resource Definition Guide* manual, for examples of how to use the DIsplay or Expand screen.

"Generic Search" - Examples

CEDA
EXPAND

LIST(*)
GROUP(*)
LIST(DF*)
GROUP(*LA)
GROUP(DFH*P)
GROUP(GROUPA)
PROGRAM(*)
GROUP(*) TRANS(TRN1)

Figure 4-18. "Generic Search" - Examples CI207.0

Notes:

- As you define more resources, it can become difficult to recall their names. The generic capability for some CEDA commands makes group and list management easier.
- Since groups and lists may be very large, expanding a subset of the group may also be helpful.
- There are two ways to specify generic names:
  a. Use * to represent any number of characters.
  b. Use **+** to represent a single character.

RDO Command Summary

- **Resource management**

– Define	– Copy
– Userdefine	– Delete
– Alter	– Rename
– Expand	– Move
– View	– Install

- **Group management**

– Expand	– Copy
– Display	– Delete
– Check	– Lock/Unlock
	– Install

- **List management**

– Expand	– Add
– Display	– Remove
– Check	– Append
	– Lock/Unlock
	– Install

Figure 4-19. RDO Command Summary CI207.0

Notes:

- The "RDO Commands" chapter in the *CICS Resource Definition Guide, SC34-6430*, describes the function and parameters of each command.
- You can abbreviate commands as long as you enter enough characters to eliminate any confusion: for example, DEF for DEFINE, or DEL for DELETE.

4.4 RDO Batch Utility DFHCSDUP

CSD Batch Utility DFHCSDUP

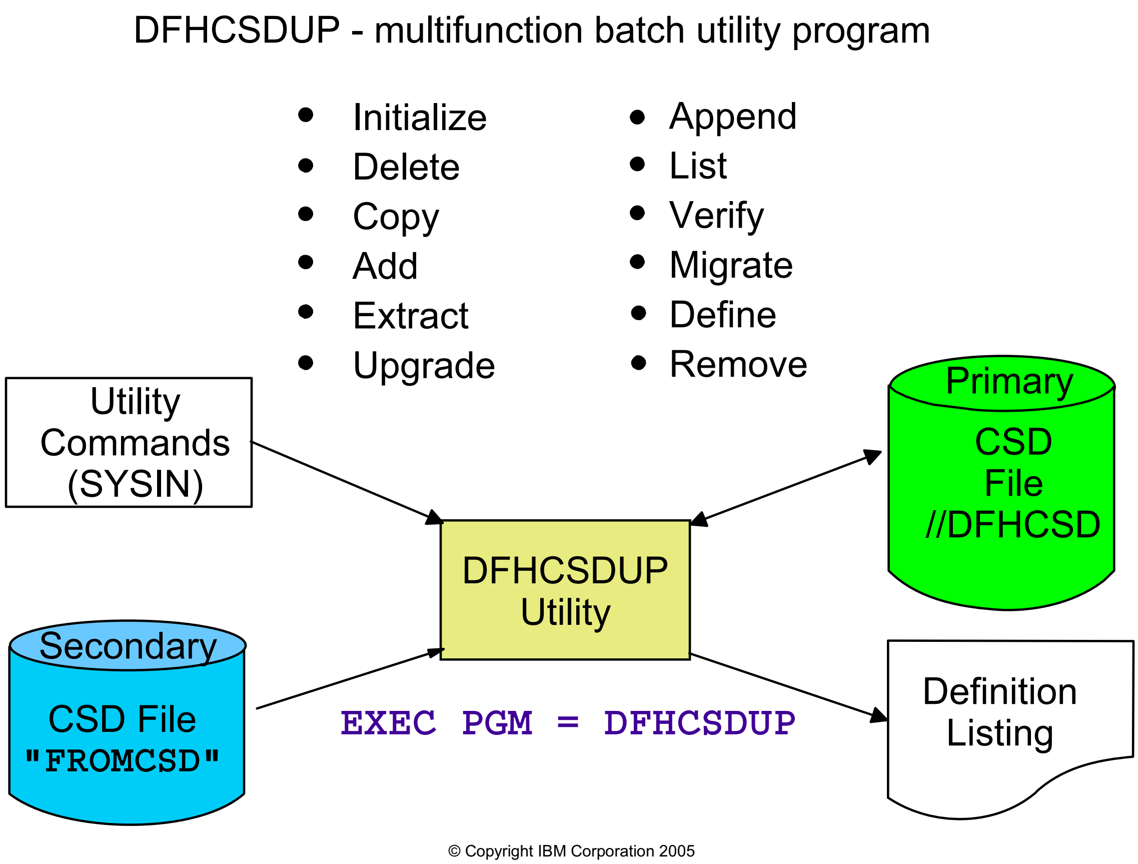


Figure 4-20. CSD Batch Utility DFHCSDUP CI207.0

Notes:

DFHCSDUP supplies similar functions to the CED* online transactions.

You cannot:

- Install resources or groups
- Copy single resources

Additional functions:

- Accessing a "secondary" CSD data set as source for copy and move operations
- INITIALIZE
- MIGRATE
- UPGRADE
- EXTRACT
- VERIFY

Details of syntax and commands of DFHCSDUP are documented in the *CICS Operations and Utilities Guide, SC34-6431*.

DFHCSDUP - INITIALIZE Function

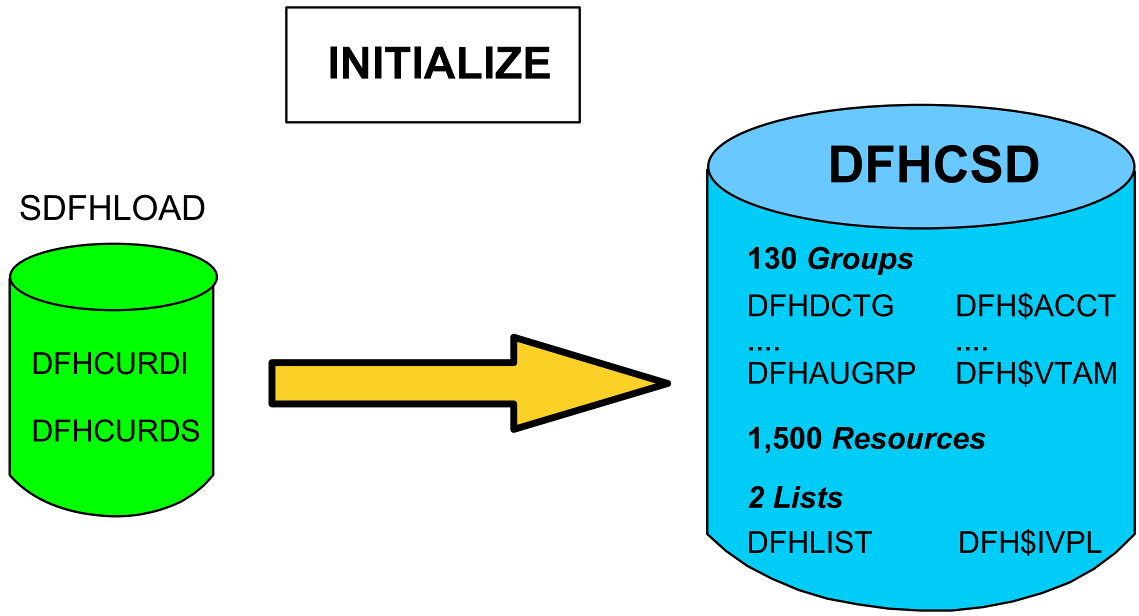


Figure 4-21. DFHCSDUP - INITIALIZE Function CI207.0

Notes:

- Any system-supplied transactions, programs (except system management nucleus programs), map sets, profiles, and sample application resources are defined within IBM-supplied RDO groups, named DFH...
- The definitions are written to the CSD data set by the INITIALIZE subfunction of the CSD batch utility DFHCSDUP. (INITIALIZE is always the first operation performed against a newly-created CSD.)
- Two lists, DFHLIST and DFH$IVPL, provide the groups needed for general system operation and the IBM-supplied installation verification respectively.
- CICS TS V2.3 added an entry for the group CEE to its DFHLIST, which is to define the LE (Language Environment) components, as the use of LE is mandatory for CICS TS V2. The CEE group contains more than 1,600 program definitions. Definition input is provided in CEESAMP(CEEDCSD).

DFHCSDUP Migrate

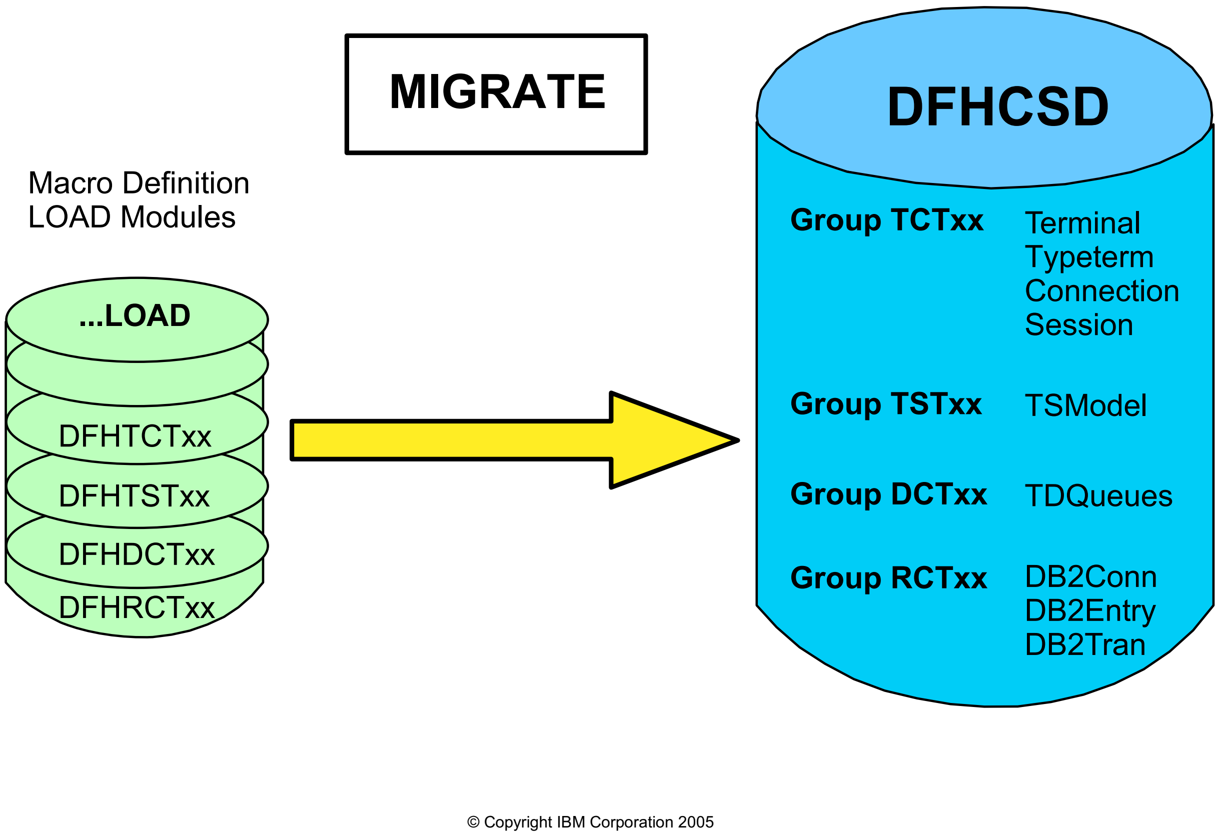


Figure 4-22. DFHCSDUP Migrate CI207.0

Notes:

- The MIGRATE keyword is used to transform macro tables into RDO definitions.
- DFHCSDUP requires macro table **load modules** as input.
- Names of RDO groups generated may be specified before migration through special macro table entries.

DFHCSDUP Upgrade

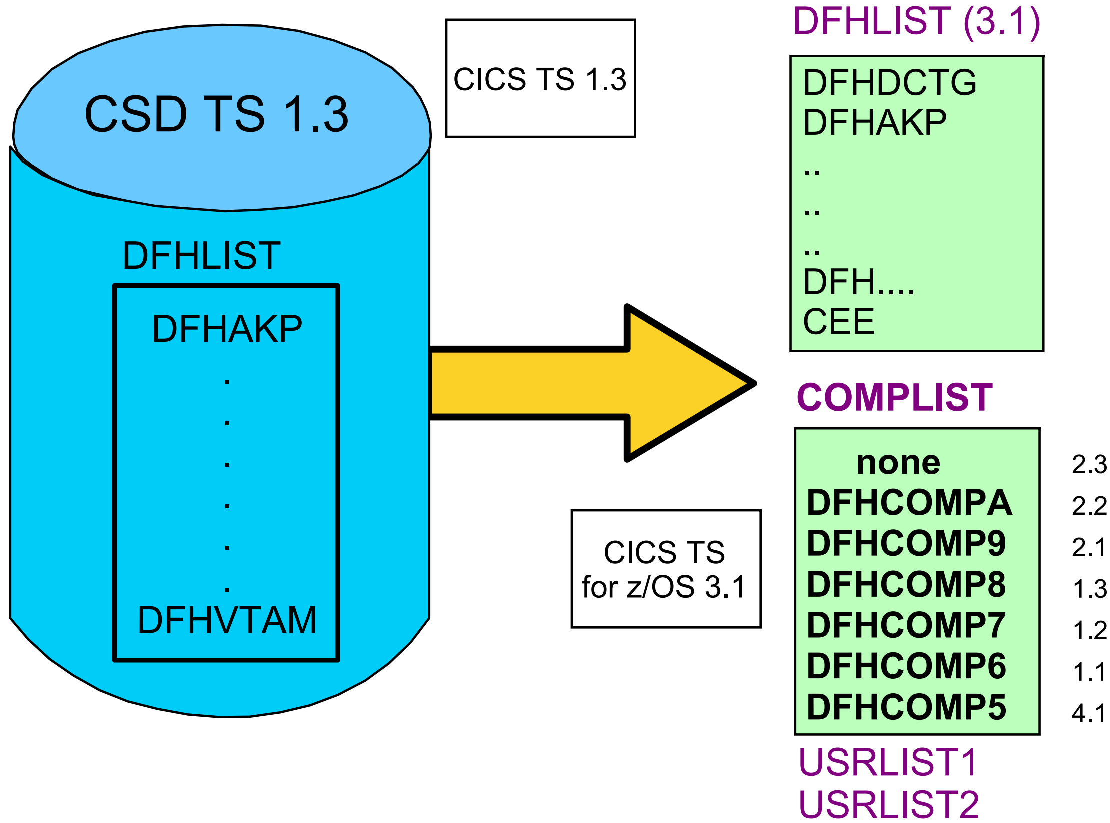


Figure 4-23. DFHCSDUP Upgrade CI207.0

Notes:

- **UPGRADE** without any further parameters is run against an existing CSD file that was initialized at an earlier release of CICS than the utility itself is running.
- Upgrading ensures that the definitions in the DFH* groups are brought up to the level of the CICS TS for z/OS function.
- Only IBM-supplied groups are affected by the `UPGRADE` command.
- Compatibility groups DFHCOMP1..9,A have to be used to allow previous releases of CICS to share an *upgraded* CSD file.
  - ***User LISTs of "old" regions have to be updated!***
  - If, after upgrading a CSD, you plan to share the CSD with earlier releases of CICS, include the appropriate DFHCOMPx compatibility groups in your start-up group list to provide the required support for earlier releases.

    The visual shows an extreme and unusual migration path, from CICS TS 1.3 to CICS TS V3.1, in order to highlight the use of the compatibility groups.

- The UPGRADE command is also used to apply any package of IBM-supplied resource definitions to the CSD file, for example when installing features on to CICS.
- Upgrading the CSD to the CICS TS for z/OS Version 2 level or later requires additional action, because the VSAM average and maximum recordsize parameters have changed.
  The CSD must now be defined with `RECORDSIZE(200 2000)`.
- Make sure that you define the CEE group which is referenced by the CICS TS 3.1 DFHLIST to your CSD file. Definition input to be processed by DFHCSDUP is provided in the LE samples library, *hlq*.CEESAMP(CEECCSD).
- One of the following methods may be used to perform this migration:

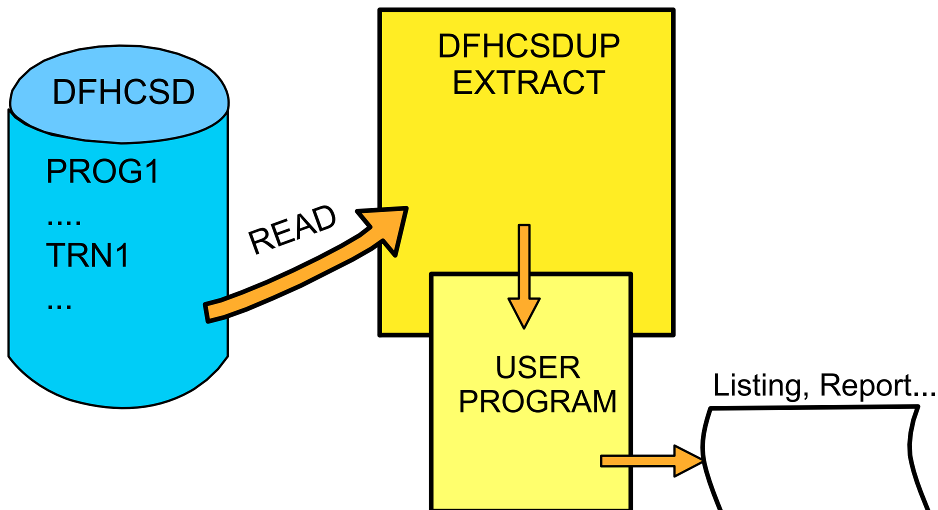
  a. Take a backup, then delete the data set, define a new one with the correct record size, and REPRO the backup into the new data set.

  b. Rename the old data set as a backup, then create a new data set, and REPRO the renamed data set into the new one.

  c. Define a new data set with the correct record size and other attributes, and then REPRO the old data set into the new one.

  Refer to the *CICS Resource Definition Guide, SC34-6430*, for details about the DFHCSDUP UPGRADE option, and to the *CICS Transaction Server for z/OS Migration Guide, GC34-642x*, for the special considerations when migrating to CICS TS for z/OS.

DFHCSDUP Extract




```
CSDUP EXEC PGM=DFHCSDUP
SYSIN DD *
 EXTRACT LI(listname)
         GR(groupname)
         OBJECTS
         USERPROGRAM(pgmname)
```


Figure 4-24. DFHCSDUP Extract CI207.0

Notes:

- **`EXTRACT`** allows a user-supplied program to process data that is read (extracted) from the DFHCSD data set and presented to that program via a documented interface.
- The user program is specified on the **`USERPROGRAM`** parameter.
- CICS supplies three types of sample user programs written in different languages (assembler, PL/I, and COBOL) as source code in the SDFHSAMP library.
- **`DFH0CBDC`** is generally considered to be the most useful one; it generates DEFINE commands for the existing resource definitions.
- For programming information about coding your own EXTRACT user programs, refer to the *CICS Customization Guide, SC34-6429*.

4.5 RDO Miscellaneous

Modifying CICS Resources

CEDB COPY G(DFHINTER) TRANS(CECI) TO(CICSMOD)

EXPAND G(CICSMOD)

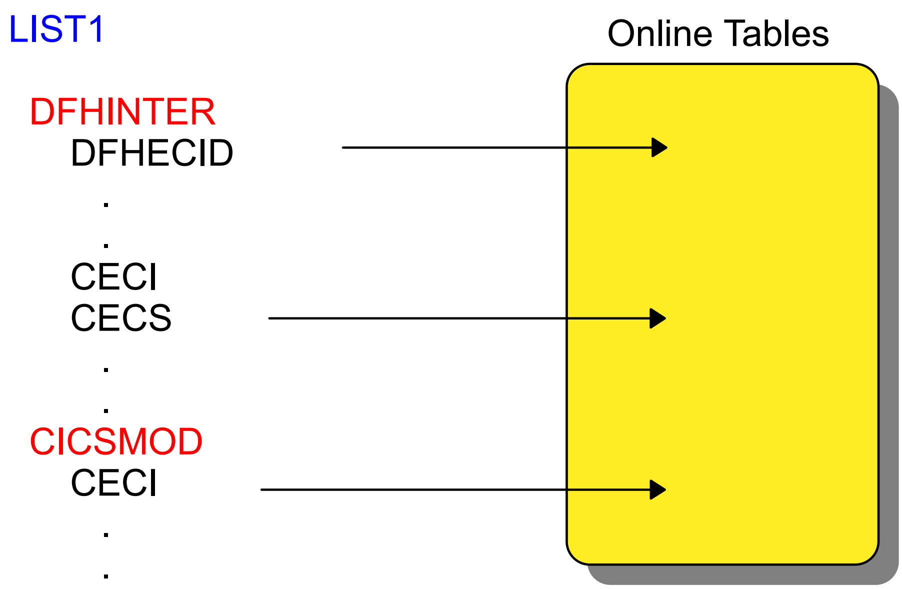


Figure 4-25. Modifying CICS Resources CI207.0

Notes:

- You are not allowed to update resource definitions in the CICS-supplied groups, neither by use of CEDA/B nor by use of the batch utility DFHCSDUP.
- If you want to security protect the command interpreter transaction (CECI), copy that transaction definition to your own group, update it, then add the new group after the IBM groups in your startup list.
- This is the main reason why CICS allows duplicate RDO resource definitions.
- Reference the “How to set up a list for initialization” section of the *CICS Resource Definition Guide, SC34-6430*, for additional information about creating lists and modifying CICS definitions.

DFHCSDUP Scan Function

SCAN TErminal(LU2)

```
DFH5633 I TERMINAL LU2  FOUND IN GROUP DFHTERM
DFH5631 I TERMINAL LU2  IN GROUP CI20TERM MATCHES
          THE IBM SUPPLIED DEFINITION IN GROUP DFHTERM
DFH5632 I TERMINAL LU2  IN GROUP EDUTERM DOES NOT MATCH
          THE IBM SUPPLIED DEFINITION IN GROUP DFHTERM
```

SCAN File(CI20FCUS)

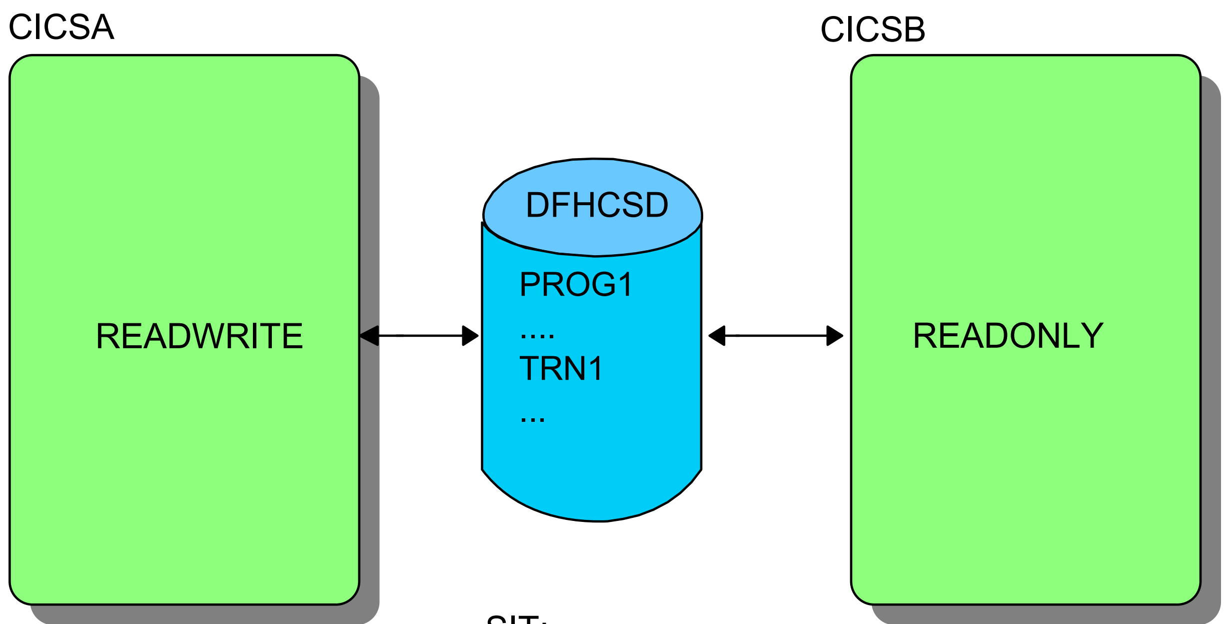
```
DFH5630 W NO IBM SUPPLIED DEFINITION FOUND
          FOR FILE CI20FCUS
DFH5633 I FILE CI20FCUS FOUND IN GROUP CI20APP2
```


Figure 4-26. DFHCSDUP Scan Function CI207.0

Notes:

- The **SCAN** option of DFHCSDUP may be used to scan all groups of a CSD data set for a specified resource.
- Its primary purpose is to identify resource definitions with names matching IBM-supplied definitions and to compare these definitions.
- The utility reports whether or not there are user definitions of the specified name and resource type, and whether or not they are equal to the IBM-supplied definition.
- If the user definition is to be compared to an IBM-supplied definition, but has been named differently, the resource name may be specified by means of the `ALIAS` parameter like this: `SCAN restype(IBMname) ALIAS(username)`
- Resource names have to be fully specified; no generic names are allowed.
- According to the second example shown in the visual, the SCAN function may be used to search for any named resources of any type. If the resource definition is not found in an IBM-supplied group, but is found in one or more user groups, a message is issued indicating the group(s) that contained it.

Sharing CSD File



SIT:

CSDACC={READWRITE|READONLY}
CSDDISP={OLD|SHR}
CSDDSN=dsname

Figure 4-27. Sharing CSD File CI207.0

Notes:

- You can have a single CSD file which you share across multiple CICS regions.
- Typically, one region, often a development region, has read/write access to allow you to update resources, while the others can only read the file. DFHCSD properties are defined in the SIT.
- When using DFHCSDUP, you can specify `READONLY` access with a JCL PARM override.
- There are additional options for sharing CSD files with regard to the new VSAM Record Level Sharing (RLS) function. These will be covered as part of the File Control unit.

Resource Definition Online (Summary)

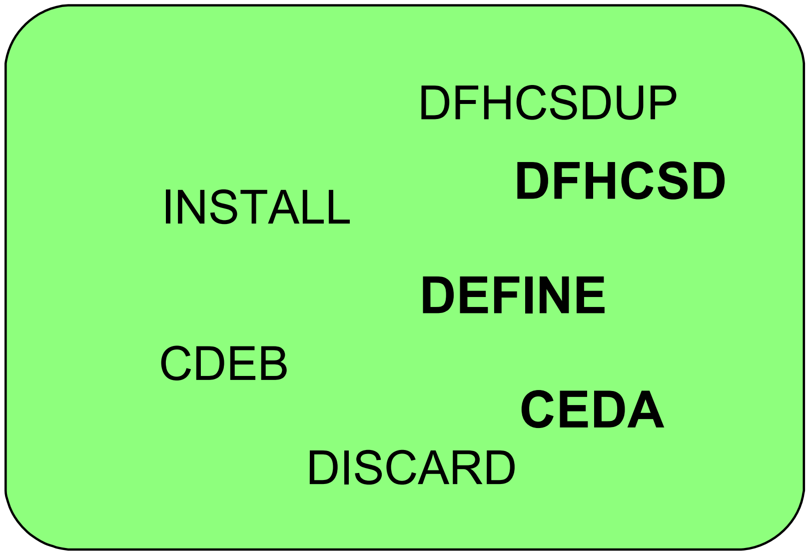


Figure 4-28. Resource Definition Online (Summary) CI207.0

Notes:

- You define most CICS resources with RDO. This facility allows you to add and modify resource definitions without interrupting the service to users.
- Use CEDB to create and maintain resource definitions in the CSD file, and CEDA to install resources into CICS storage.
- You define resources as part of groups; a group may be associated to one or more lists. Lists are used to designate which resources are to be used in a CICS execution.
- You manage the CSD file with the VSAM IDCAMS utility and the CSD Utility Program DFHCSDUP.
- The DFHCSD file may be shared between several CICS TS regions, possibly all regions of a sysplex. This is to minimize definitions and avoid multiple occurrences of the same data, with its negative impact on administration.
- Using the `EXEC CICS CREATE` command, resources may be installed without placing their definitions on the DFHCSD file. This command is used by CPSM to supply a single point of definition for CICS TS systems associated with a “CICSPlex”.

Unit Summary

Having completed this unit, you should be able to:

- Identify resources that can be defined online
- Explain the relationship between resources, GROUPs, and LISTs
- Manage the resources in GROUPs and LISTs
- Identify which resource definitions CICS loads at initialization
- Dynamically install resource definitions while CICS is active
- Manage the RDO data set

Figure 4-29. Unit Summary CI207.0

Notes:

Unit 5. Supporting Terminal Access

What This Unit Is About

This unit teaches the concepts and “to do”s of supporting the direct logon of user terminals to CICS over the network, which is the traditional way of accessing CICS.

Whenever we talk about “terminals” in this context, printers and consoles are included, as CICS considers and handles them as special terminal devices.

The unit is made up of four lecture parts (topics); thereafter a lab exercise will provide an opportunity to apply some of the terminal definition parameters introduced and discussed throughout the lectures.

The first topic shows you the concepts of defining 3270 VTAM terminals (“dumb terminals”) to CICS TS, which includes support for connection of workstations using 3270 terminal emulation products.

The second topic covers the CICS TS autoinstall function that is certainly in use by most customers, at least for these simple terminals.

The topic that covers handling of terminal definitions within the RDO facility is positioned as the third one. Additionally, some typical runtime issues are addressed here.

We will close this unit by discussing some useful facts about CICS TS and VTAM interactions, where some additional features have been added throughout the latest releases.

What You Should Be Able to Do

After completing this unit, you should be able to:

- List the network information necessary to define terminals
- Explain the relationship between TERMINAL and TYPETERM definitions
- Define 3270 terminals supported by VTAM
- Explain the concepts of autoinstall
- Code SIT and VTAM parameters related to communications

How You Will Check Your Progress

- Machine lab.

References

CICS System Definition Guide, SC34-6428

CICS Resource Definition Guide, SC34-6430

Unit Objectives

After completing this unit, you should be able to:

- List the network information necessary to define terminals
- Explain the relationship between TERMINAL and TYPETERM definitions
- Define 3270 terminals supported by VTAM
- Explain the concepts of autoinstall
- Code SIT and VTAM parameters related to communications

Figure 5-1. Unit Objectives CI207.0

Notes:

5.1 Basic Concepts of CICS Terminal Definition

Terminal Representation

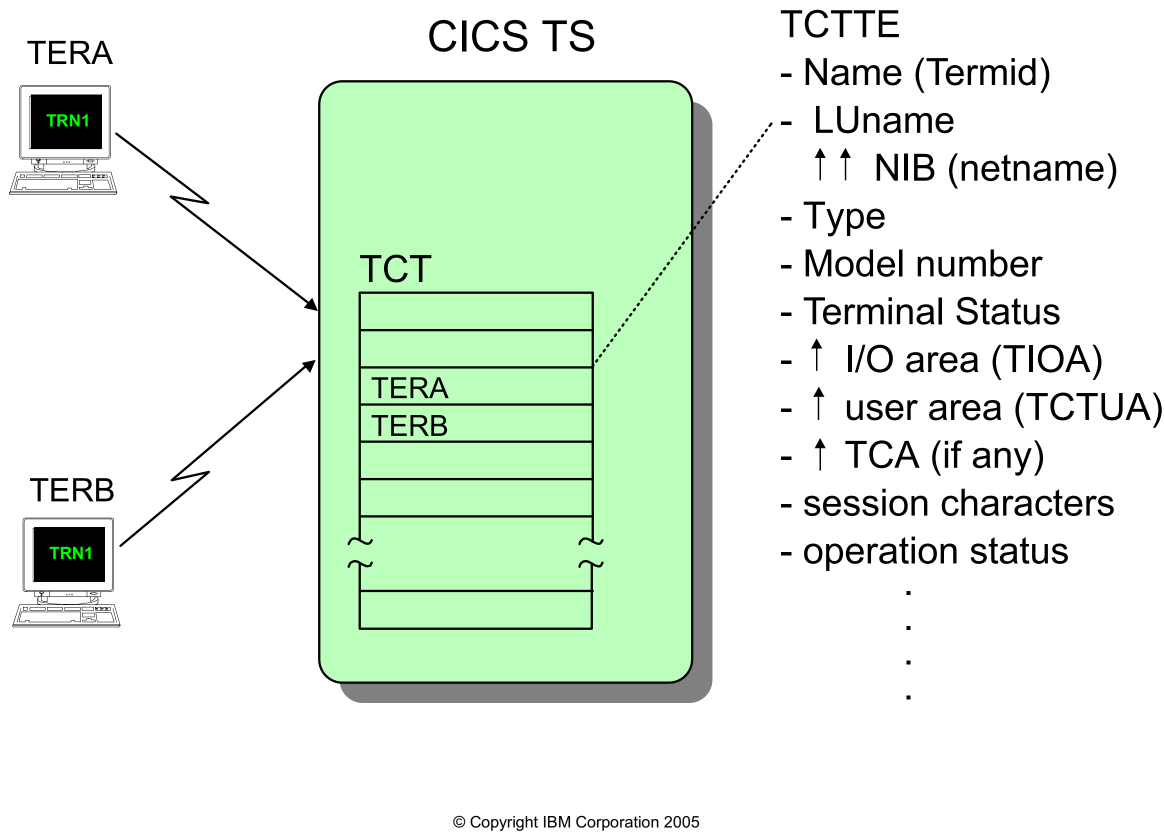


Figure 5-2. Terminal Representation CI207.0

Notes:

- When a terminal enters a session with CICS TS, an entry in the internal Terminal Control Table (TCT) representing this terminal will be there, which either has been predefined or is created automatically at logon time.
  - The TCT entries for terminals are called TCT Terminal Entries (TCTTE).
  - This control block describes the terminal with all its features and capabilities, how it has to be handled by CICS, and contains dynamic information such as status flags and pointers to other system areas.
  - The static part of the TCTTEs is created based on RDO terminal definitions.
- CICS tasks (transactions) that were initiated from a terminal are “owning” these terminals exclusively.
  - The terminal is called the task's **principal facility**.
  - There may be only one task running against a given terminal at a certain time.
  - CICS controls a connected terminal’s operation mode. During task execution, the terminal is “blocked”, until CICS has delivered a response and freed the terminal.

Terminal Management

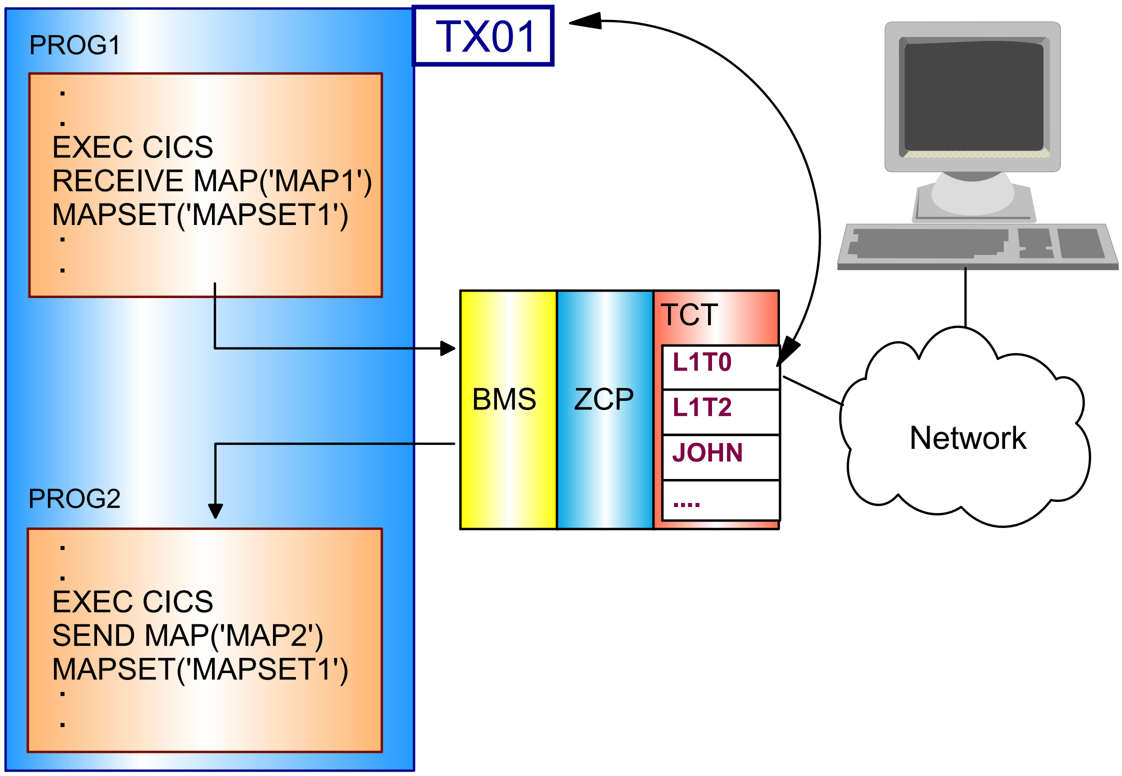


Figure 5-3. Terminal Management CI207.0

Notes:

- Applications invoke terminal control service by means of the (EXEC CICS) SEND and RECEIVE commands.
- These commands are always directed against the task's principal facility, in other words, the terminal that initiated the appropriate transaction.
- BMS (Basic Mapping Support) is the CICS function that provides the user interface for applications, mapping the raw application data with application-defined screen layouts called “map sets”. BMS is covered by Application Programming courses.
- DFHZCP is the CICS VTAM terminal control program. It supports VTAM devices and the use of z/OS consoles as CICS terminals. Definitions for ZCP-supported devices have to be provided via RDO.
- The original CICS terminal control program, still there, is (DFH)TCP. It supports a TCAM DCB network, and also sequential terminals such as Card Reader/Line Printer (CRLP). These devices must be defined by macros. Documentation for macro terminal definitions is in the *CICS Resource Definition Guide, SC34-6430*.

RDO Terminal Definition

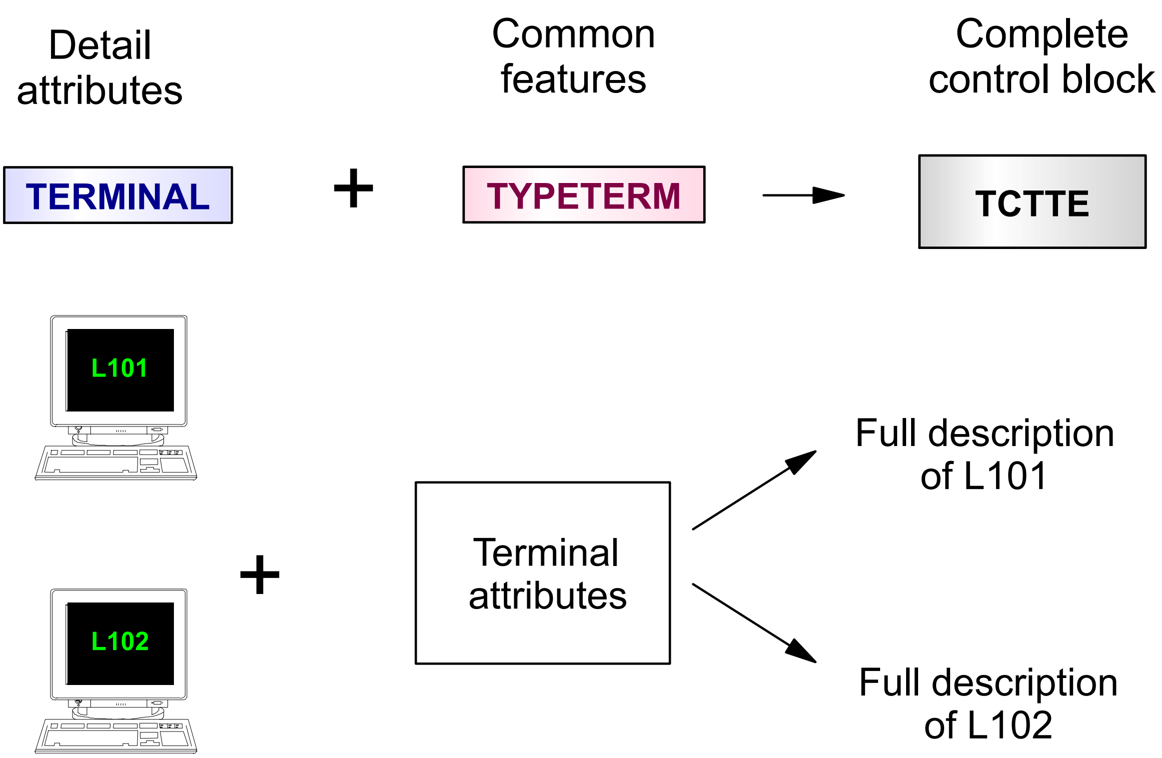


Figure 5-4. RDO Terminal Definition CI207.0

Notes:

- Terminal attributes are features common to a set of terminals. When combined with detail terminal attributes, the result is a full terminal definition.
- Terminal definition and maintenance are simplified, since these general characteristics are not repeated for each terminal.
- The **DEFINE TYPETERM** command creates a resource containing features common to many terminals, such as screen size, alarm, lightpen, and diagnostic display options.
- You define individual terminals with the **DEFINE TERMINAL** command, specifying such attributes as terminal name, VTAM netname, priority, and security.
- Each TERMINAL definition must refer to a TYPETERM.
- Generally, only a few TYPETERM definitions will be needed, as one TYPETERM definition may be referred to by many TERMINALs, if your terminal network has a small number of unique sets of common characteristics.

TYPETERM Definition (Main Parameters)

CEDA DEFINE TYPETERM(name) GROUP(name)

DEVICE	3270	--Display	local devices (non-SNA)
	3270P	--Printer	
	CONSOLE	--Console	
	LUTYPE2	--Display	SNA remote and local devices
	LUTYPE3	--Printer	
	SCSPRINT	--SCS-Printer	
DEFSCREEN	rows,cols		
QUERY	NO|COLD|ALL		
SHIPPABLE	NO|YES		
UCTRAN	NO|YES|TRANID		

Figure 5-5. TYPETERM Definition (Main Parameters) CI207.0

Notes:

- TYPETERM parameters are described in the “RDO resource Types and Their Attributes” chapter of the *CICS Resource Definition Guide, SC34-6430*.
- **DEVICE** specifies the type of terminal. This parameter is crucial for successful support of the facilities to be connected. Default values for many other attributes depend on this parameter.

  Before defining terminals, and providing their TYPETERM definitions, you need to know:

  - Whether the terminal is attached to an SNA or a non-SNA controller
  - Whether the terminal is a display or a printer, and for SNA printers, if it is:
    - 3270 data stream, or
    - SNA Character String (SCS).
  - The VTAM mode table entry the Logical Unit (LU) is using; this is meaningful for automatic installation of terminals.
  - Which z/OS consoles can communicate with this CICS region.

- **DEFSCREEN** specifies the default screen size to be applied to a terminal. CICS generates its SNA bind from this information.
- **QUERY** specifies whether CICS should use the QUERY structured field to determine certain characteristics ("device properties") of a terminal, so they may be left undefined. Use QUERY when available.
- **SHIPPABLE** specifies whether the definition is allowed to be sent to a remote system if this device tries to initiate a remote transaction.
- **UCTRAN** controls whether or not input entered from a terminal is translated to uppercase.

TERMINAL Definition (Main Parameters)

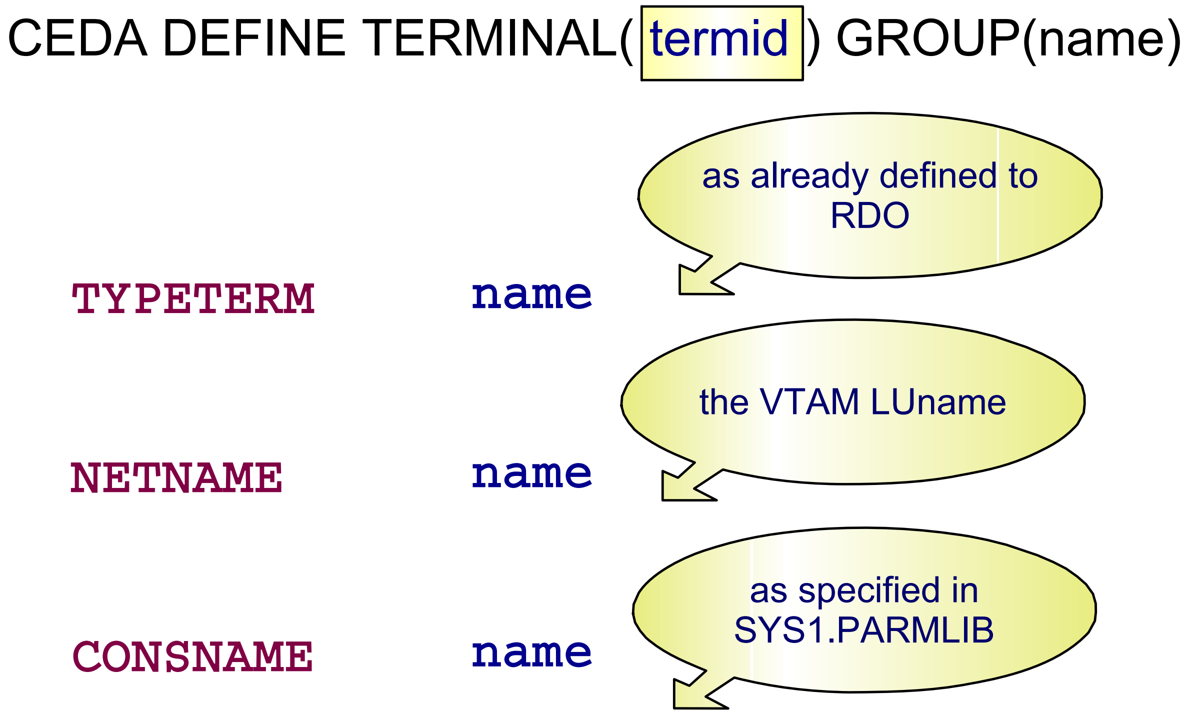


Figure 5-6. TERMINAL Definition (Main Parameters) CI207.0

Notes:

The *CICS Resource Definition Guide, SC34-6430*, describes the TERMINAL parameters.

- Use a terminal definition for each terminal you want to define explicitly.
  This will build up the TCT Terminal Entry (TCTTE) when installed.
- **TERMINAL** name is the 4-character CICS terminal name, which is called the TERMID within CICS.
- **NETNAME** is the name by which the terminal is known to the network. VTAM folks will call it the terminal's LUNAME.
- For a z/OS **console** to be used as a CICS terminal, specify its name as defined in SYS1.PARMLIB member, CONSOLnn. To define a TSO user as a console device, specify CONSNAME(name), where name is the TSO userid.
- The **TYPETERM** parameter names the resource containing the common characteristics used to complete the terminal definition.

CEDA Check

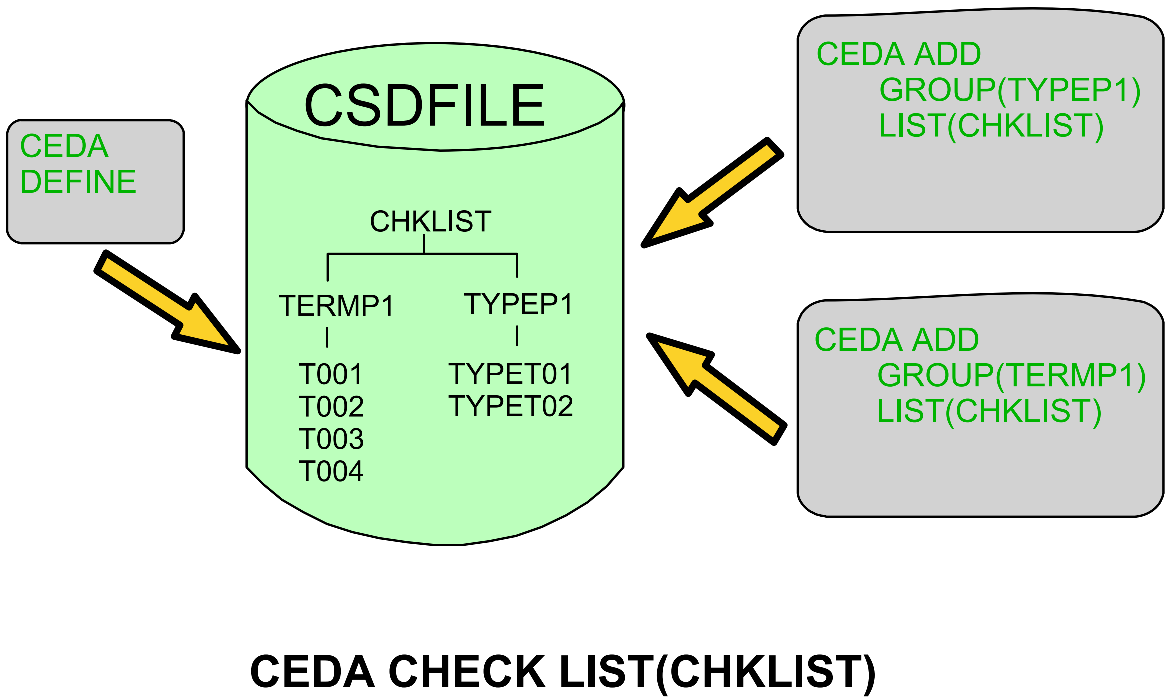


Figure 5-7. CEDA Check CI207.0

Notes:

You can use the CHECK command to check a group of TERMINAL definitions (it resolves references from display devices to printers, for instance). It is not very useful in resolving TYPETERM references if these are in a separate group because it would produce many unwanted messages for missing TYPETERMs.

To avoid duplicating TERMINAL names, you could maintain a list of all groups containing TERMINAL definitions. You can use CHECK LIST to ensure that all new TERMINAL names are unique.

All related resources you are checking must be in the same group or list.

Would a check of the group TERMP1 have errors?

Would a check of the group TYPEP1 be successful?

Installing Terminals

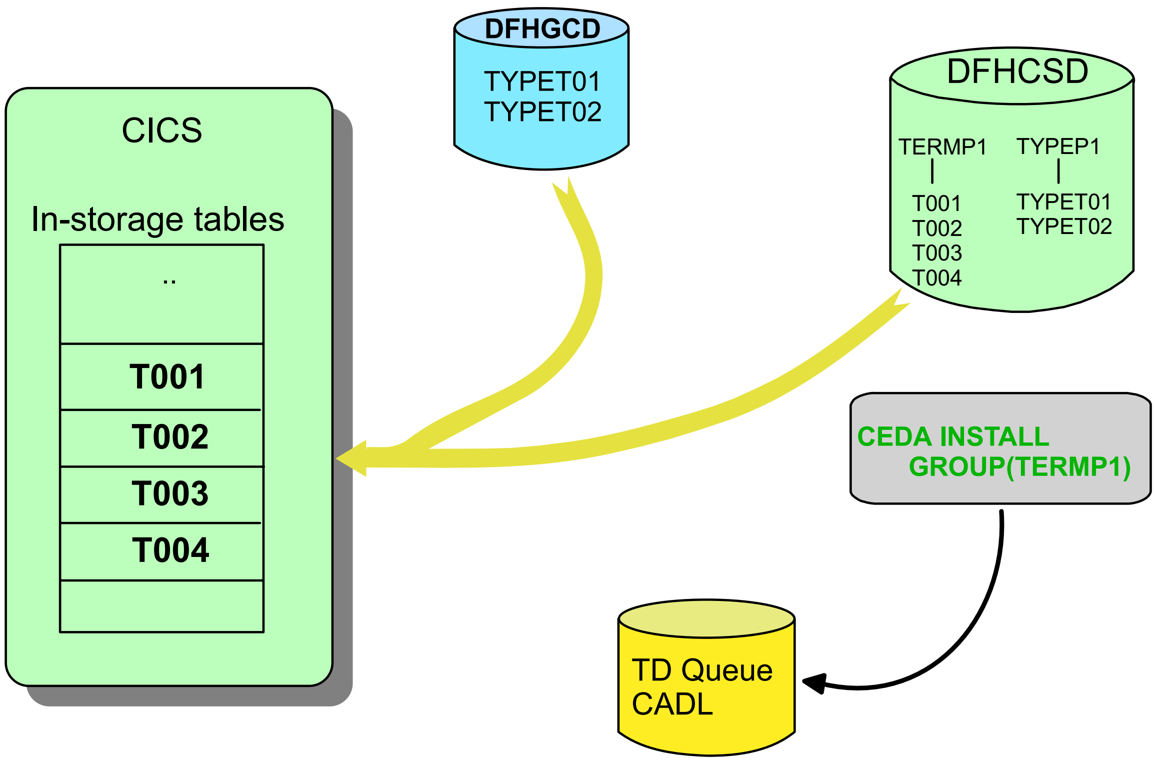


Figure 5-8. Installing Terminals CI207.0

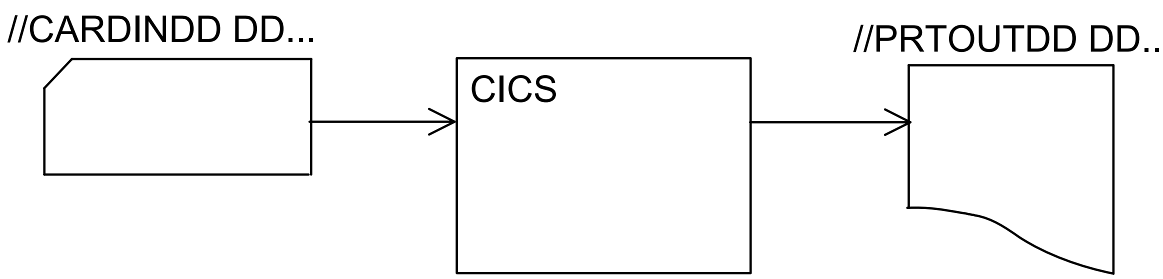
Notes:

- At **TERMINAL** installation time, the **TERMINAL** and **TYPETERM** definitions are merged to create a TCTTE.
- A **TERMINAL** can only be installed if the referred **TYPETERM** is installed at the same time or has been installed before.
- So, to add terminal definitions to an active CICS, you would:

  a. Install the TYPETERMs

  b. Install the TERMINALs
- TYPETERMS are installed in the global catalog (DFHGCD); no change is made to active TCTTEs that reference TYPETERMs which may be reinstalled.
- CEDA activities are “logged” for documentation into a TD Queue named CADL.

Card Reader/Line Printer



DFHTCT TYPE=....,

Figure 5-9. Card Reader/Line Printer CI207.0

Notes:

CICS supports transaction input provided in a sequential data set (may be a member of a PDS) which is treated as if this input was sent by a terminal. The same is true for output created by an application. This option is called *sequential terminal support*.

- The data sets used for this kind of operation have to be represented by TCT terminal definitions (no RDO support). There, a DDNAME is specified, which has to appear in the CICS startup JCL specifying the real data set name.
- Input is read immediately after control is given to CICS, and thus may be used to perform initialization routines.
- Each transaction in the input stream is terminated with the value defined by the EODI parameter in the SIT. The default value is the backslash.

```
//CARDINDD  DD  *
INQY 123456\
CEMT P SHUT\
```

For details, refer to the *CICS System Definition Guide*.

5.2 Terminal Autoinstall

Why Terminal Autoinstall?

✓ **Reduce/minimize administration**

✓ **Save virtual storage**

✓ **Support new users, new terminals, or both**

✓ **Support changing terminal characteristics**

Figure 5-10. Why Terminal Autoinstall? CI207.0

Notes:

- Infrequently-used terminals that have been installed still use virtual storage, even when not in use.
- When using autoinstall, dynamic storage for the terminal definition is used only from the time the terminal logs on until it logs off.
- New terminals have access to CICS.
- Terminal characteristics change, for example screen size.

Reference the "Autoinstall for Terminal Definitions" chapter of the *CICS Resource Definition Guide, SC34-6430*, for a description of the autoinstall process.

Logon Processing

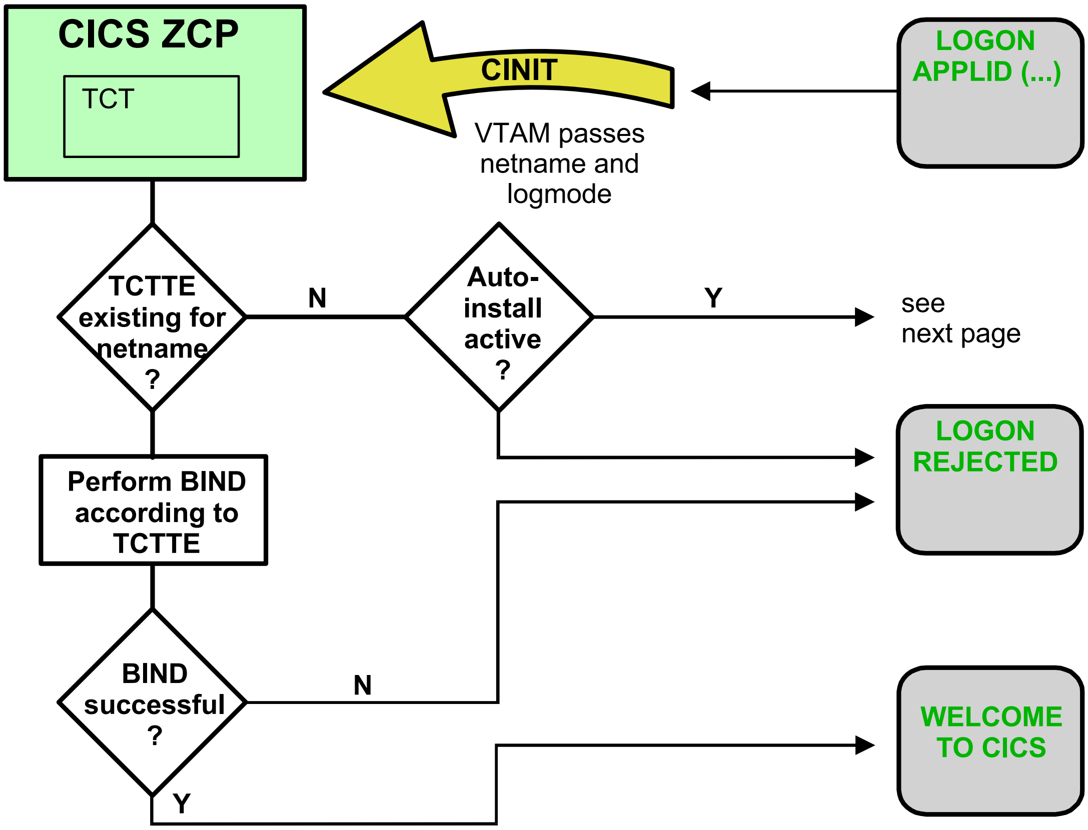


Figure 5-11. Logon Processing CI207.0

Notes:

Terminal logons are always processed in the following way:

1. VTAM passes the NETNAME and VTAM logon mode table data to CICS (CINIT RU).
2. CICS ZCP scans its in-storage TCT in order to find an entry with the NETNAME of the logon requesting terminal.
3. If an entry is found, CICS builds the VTAM BIND data according to the terminal definition (TCTTE). CINIT data supplied by VTAM is disregarded.
   - If this definition is correct, the BIND will succeed and the logon will be completed.
   - If this definition is incorrect, the BIND will fail and the logon will be rejected.

Autoinstall Processing

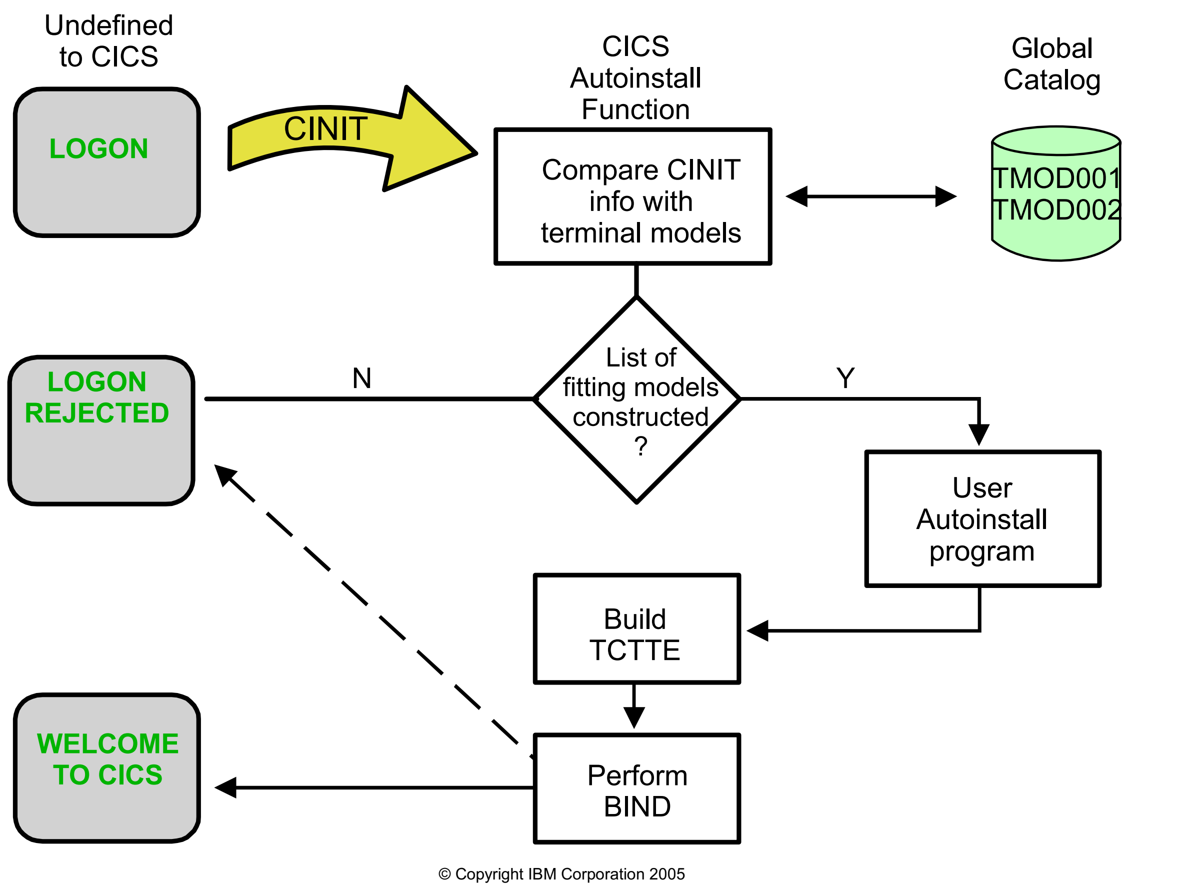


Figure 5-12. Autoinstall Processing CI207.0

Notes:

- Autoinstall processing is only initiated for terminals **not defined** to CICS.
- The autoinstall function depends on the existence of “fitting” models.
- Selection of the correct model depends on the validity of the VTAM logon mode table information.

Note the following facts:

- CICS will always scan its TCTTE for an existing entry naming the netname of the logon requesting terminal.
- A terminal that is defined to CICS with its netname will never be handled by autoinstall!

Autoinstall Models

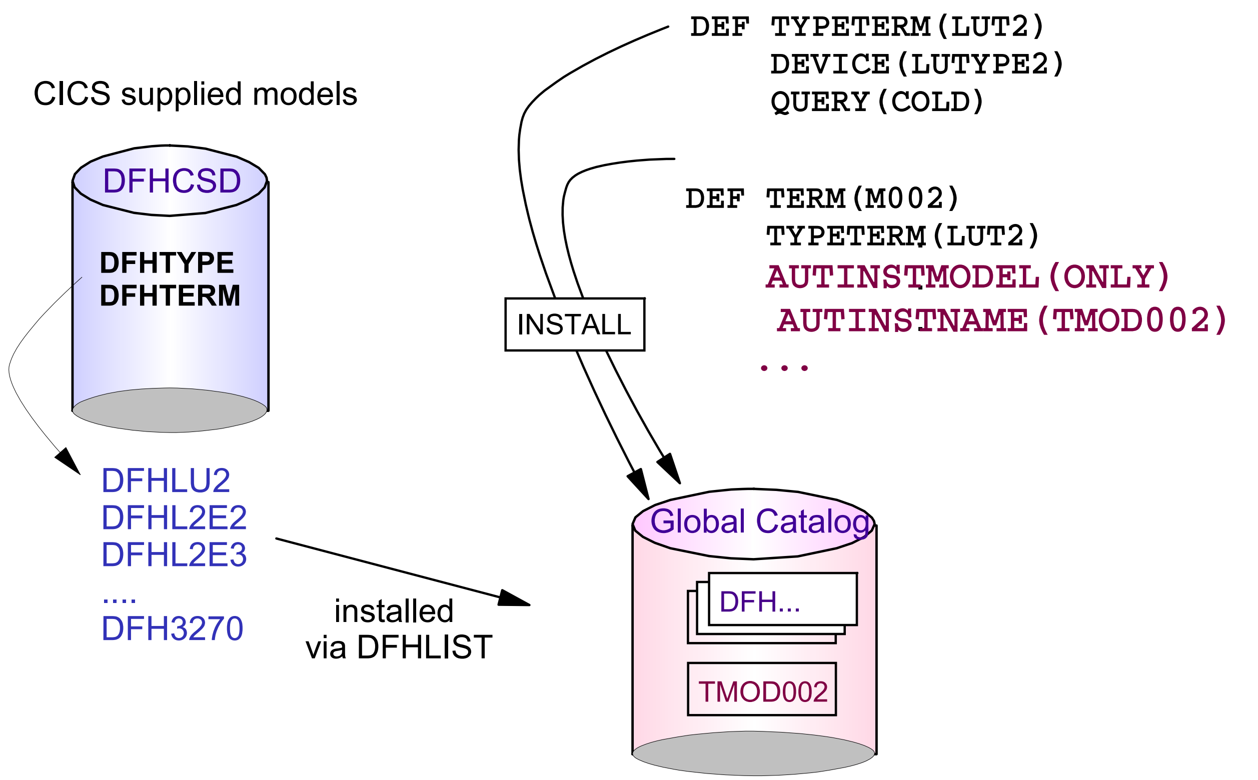


Figure 5-13. Autoinstall Models CI207.0

Notes:

- IBM-supplied group DFHTERM contains model definitions (referring to TYPETERMs defined in group DFHTYPE) that match with VTAM default logmode definitions.
- You can define your own models to support special requirements within your installation.
- Model terminals differ from a normal terminal in the following ways:
  - `AUTINSTMODEL(YES or ONLY)` is coded
  - `AUTINSTNAME` is the model name passed to the user autoinstall program and it appears in autoinstallation message DFHZC6935.
- Issue a CEDA INSTALL for the model terminal.
  For a terminal defined with AUTINSTMODEL(ONLY), this places a model in the global catalog.
  With `AUTINSTMODEL(YES)`, both a terminal definition and a model are created.
  To avoid confusion, you should code AUTINSTMODEL as ONLY and keep all models in one group.
- Existing model names can be inquired using `CEMT INQ AUTInstmodel`
- Model terminals are **not** displayed on `CEMT INQ TERMINAL`!

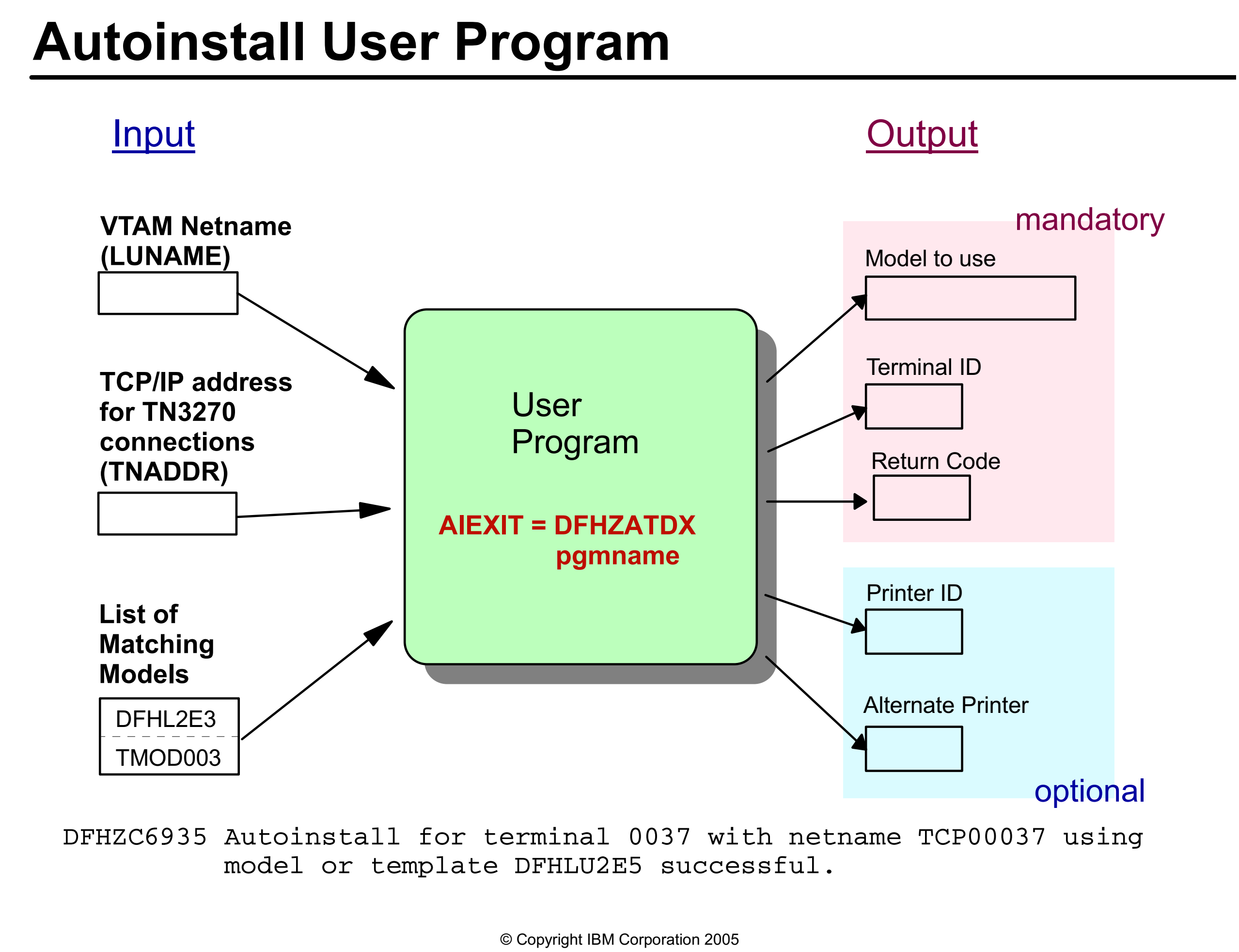


Figure 5-14. Autoinstall User Program CI207.0

Notes:

- The user program may determine the required output information by any means and criteria. It may issue any API command (except SEND/RECEIVE), but exit designers should be aware of the performance impact that would take effect for every single logon.
- The default program supplied by CICS selects the first model in the list and uses the last four non-blank characters of NETNAME as the terminal identifier.
  Note that CICS system code checks for active terminal models in alphabetical order, by their names, so the first entry in the list need not be the most suitable.
- Using VTAM's Model Terminal Support (MTS) feature, you can define in your VTAM definitions both the terminal model name and CICS printer TERMIDs that CICS should use for autoinstall. These definitions are honored by the default exit program.
- Each terminal autoinstall event is accompanied by system messages as follows:

  ```
  DFHZC6907 I 02/18/2004 09:01:34 CI20DE0 Autoinstall starting for netname
  FREH8939. Network qualified name is GBIBMFR.FREH8939.
  ```



```
DFHZC5966 I 02/18/2004 09:01:34 CI20DE0 INSTALL started for TERMINAL (
8939) (Module name: DFHBSTZ).
DFHZC6935 I 02/18/2004 09:01:34 CI20DE0 Autoinstall for terminal 8939
with netname FREH8939 using model or template DFHL2E2 successful.
```

- Starting with CICS TS for z/OS Version 2 Release 3, CICS provides code in the sample user exits to extract the TCP/IP address of terminals connecting through Telnet 3270.

  For these terminals, the autoinstall function issues an additional system message which looks like this:

```
DFHZC6908 I 02/18/2004 10:10:04 CI20DE0 Autoinstall in progress for
netname A1Z8U063. TN3270 IP address is 10.12.149.2:1411.
```

- Your autoinstall program also gets control when the terminal definition is deleted. This event is also reported in a system message:

```
DFHZC5966 I 02/18/2004 10:00:15 CI20DE0 DELETE started for TERMINAL (
8939) (Module name: DFHBSTZ).
```

- Full documentation for the autoinstall program is in the “Writing an Autoinstall Control Program” chapter of the *CICS Customization Guide, SC34-6429.*

Controlling Autoinstall

SIT:

```
AILDELAY=0 | hhmmss
AIRDELAY=700 | hhmmss
AIQMAX=100 | number
AIEXIT=DFHZATDX | pgmname
```

CEMT INQ
AUTOINSTALL

CEMT SET AUTOINSTALL
MAXREQS() PROGRAM()

Figure 5-15. Controlling Autoinstall CI207.0

Notes:

AILDELAY Specifies how long CICS will wait after the user logs off before deleting an autoinstalled terminal from in-storage tables.
The default is zero, which means that CICS (namely the default AI exit) deletes autoinstalled terminal at logoff with no delay.

AIRDELAY Specifies the handling of autoinstalled terminal control blocks in the event of an emergency restart.

If AIRDELAY is not zero, an autoinstalled terminal is reinstalled after an emergency restart. If no session is started for this terminal before the expiration of AIRDELAY, the terminal entry will be deleted.
The default is 7 minutes.

AIQMAX Specifies how many pending requests for autoinstall may be queued before CICS would reject any further ones.
`AIQMAX=0` turns off autoinstall.

Console Support and Autoinstall

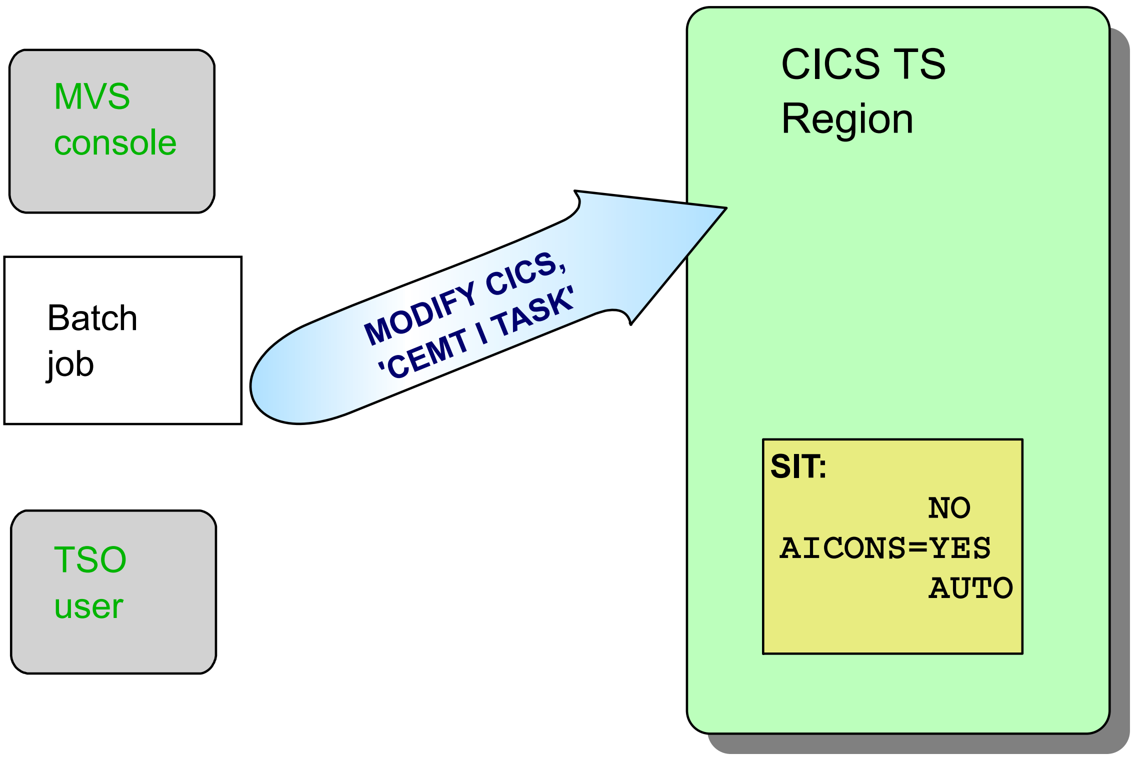


Figure 5-16. Console Support and Autoinstall CI207.0

Notes:

- z/OS consoles defined as CICS terminals can communicate with CICS. The z/OS **MODIFY** command is used for this communication.
- Assuming that the permissions required are there, a TSO user can enter MODIFY commands either from SDSF or in the following way, if the userid is specified as the console name in the terminal resource definition `(CONSNAME(tsouser))`.

  ```
  CONSOLE MODIFY cicsid,'command'
  ```

- The SIT parameter, `AICONS`, controls autoinstall for consoles with the following options:

AICONS=YES Console autoinstall is handled the same way as VTAM terminal autoinstall, by use of the autoinstall control program.

AICONS=AUTO Console autoinstall is done by means of the first retrieved console model, without calling the autoinstall control program. Normally, the `CONS` terminal from the DFHTERM group will take effect.

AICONS=NO No autoinstall; only predefined consoles can contact CICS.

Terminal INQUIRE and SET

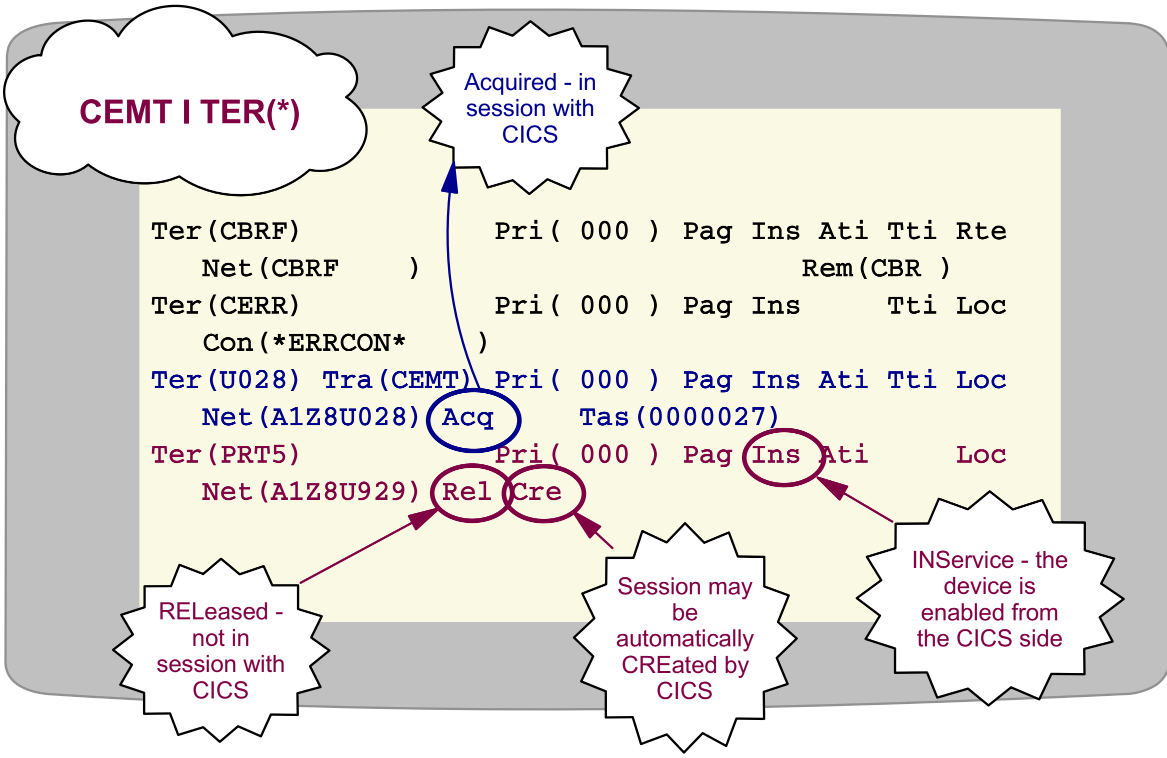


Figure 5-17. Terminal INQUIRE and SET CI207.0

Notes:

Terminals known to CICS may be inquired and set by use of the CEMT master terminal transaction.

- From the information returned on `CEMT INQUIRE`, it cannot be determined whether a terminal has been predefined or autoinstalled.
- Overtyping the appropriate status information may cause a terminal to connect or disconnect, as requested.
  SETting attributes does not affect the appropriate resource definition.

The so-called create bit is an important indication for printer devices. It is initially based on the `CREATESESSION` parameter in the typeterm object, but may be switched off by CICS if a session with a printer cannot be established, for example because the device is powered off.

5.3 CICS TS and VTAM Relationships

Defining CICS to VTAM

```
APPL1  VBUILD TYPE=APPL

A4OASICT APPL    ACBNAME=CI20GR1,
                 AUTH=(ACQ,PASS,VPACE),
                 VPACING=5,
                 EAS=100,
                 PARSESS=YES,
                 HAVAIL=YES,
                 SONSCIP=YES,
                 MODETAB=MTD1SMID,
                 DLOGMODE=#INTER,
                 APPC=NO,
                 PERSIST=MULTI,
                 LUAPFX=T1
```


Figure 5-18. Defining CICS to VTAM CI207.0

Notes:

- Network definitions required by VTAM applications are defined in **SYS1.VTAMLST**. The VTAM network name, which is also called the VTAM **APPLID**, is given by the label of the APPL statement A4OASICT in the sample shown above.
  This name has to be unique within the complete VTAM network.
- An “application minor node” may be coded by use of the **`ACBNAME=`** parameter. This name must be unique at the VTAM domain level, and therefore may be more descriptive than the major node name.
  If this parameter is omitted, it defaults to the VTAM network name.
- For each CICS region to be started, there has to be one APPL definition.
- The parameters marked with an asterisk are referred to in detail in the following foils.
- Basically, this definition is not CICS release-dependent. But it may specify options that are only supported by certain releases.
- Refer to section “VTAM Definitions Required for CICS” in the *CICS Transaction Server for z/OS Version 3.1 Installation Guide, GC34-6426*, for a description of these parameters.

Interdependent Parameters

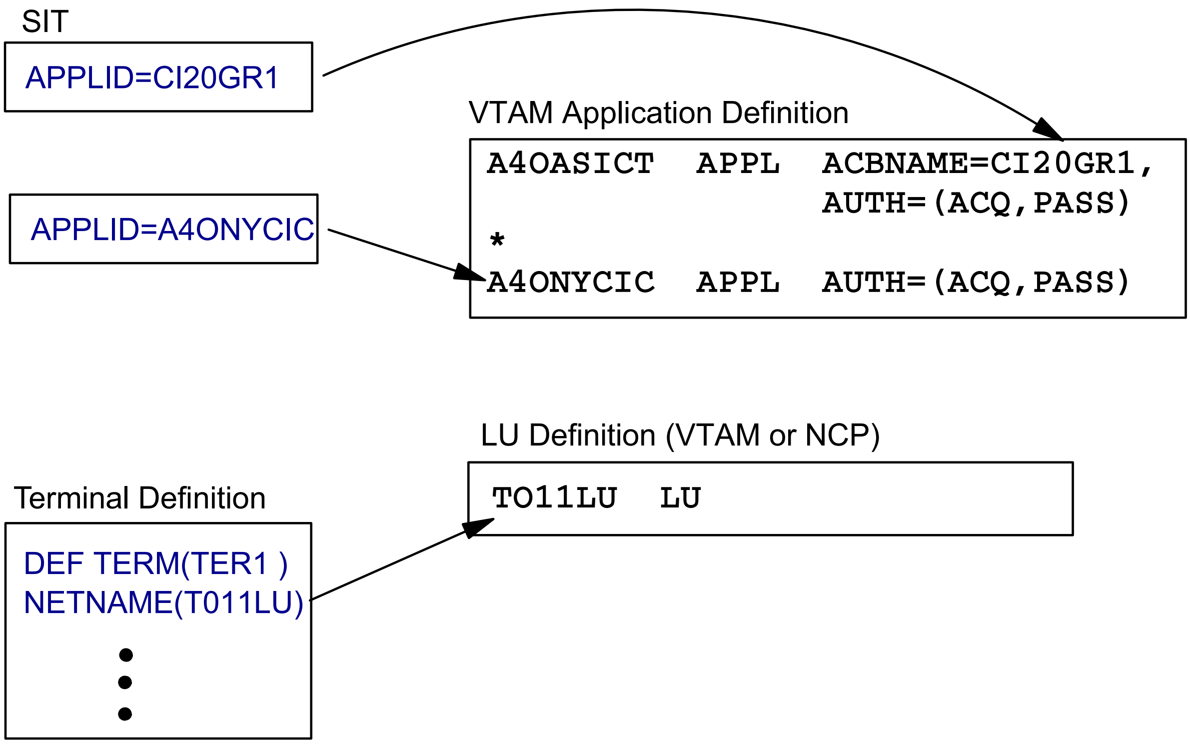


Figure 5-19. Interdependent Parameters CI207.0

Notes:

- When CICS TS is started, as with every VTAM application, it will call up to VTAM with the `OPEN ACB` function.
  - Thereby CICS has to specify the name by which it is known to VTAM, namely its **ACBNAME,** which defaults to the unique network name, if not specified explicitly.
  - Therefore, this name has to be defined to CICS at startup. This is done by the `APPLID=` parameter in the SIT.
- For terminals that are to be predefined to CICS, the **network name** (netname) for each terminal, defined in the **TERMINAL** resources, must match the name of the **VTAM LU** statement for this terminal.

The HPO Option and Receive-Any Processing

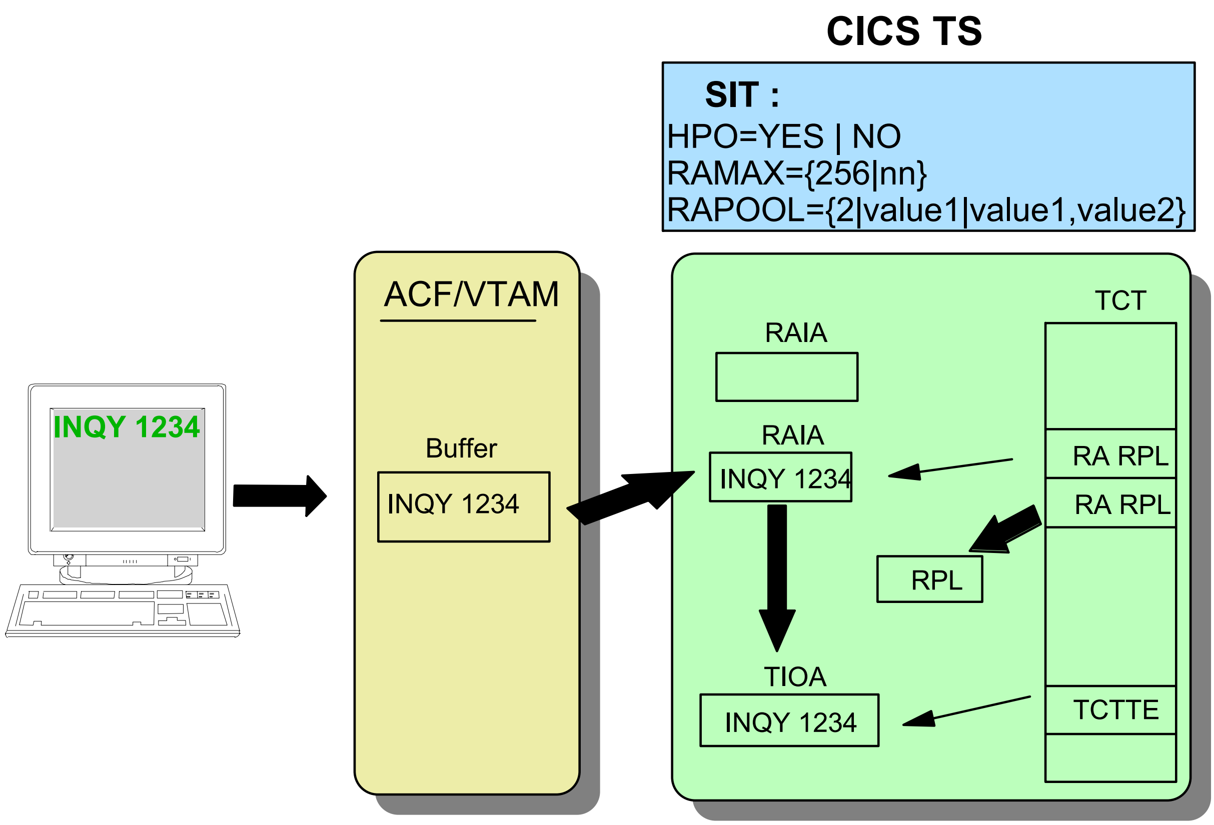


Figure 5-20. The HPO Option and Receive-Any Processing CI207.0

Notes:

- The **high performance option** (HPO) bypasses some of the validation functions performed by z/OS on I/O operations, and supports parallel scheduling of VTAM-related processing, thus reducing the path length of VTAM operations in a multiprocessor environment.
- **RAMAX** is the size of each Receive Any Input Area (RAIA) buffer that is used to receive the first terminal input for a transaction from VTAM.
  - For most terminals, the default value of 256 is sufficient.
- **RAPOOL** specifies the number of concurrent receive-any requests that CICS is to process from VTAM, with a preallocated RAIA buffer allocated to each of them.
  - The first value applies to non-HPO systems, the second for HPO systems.
  - CICS determines the actual number of RECEIVE ANYs dynamically, based on the effective RAPOOL value and the number of active tasks.
  - For HPO systems, usually a small value (<=5) is sufficient.

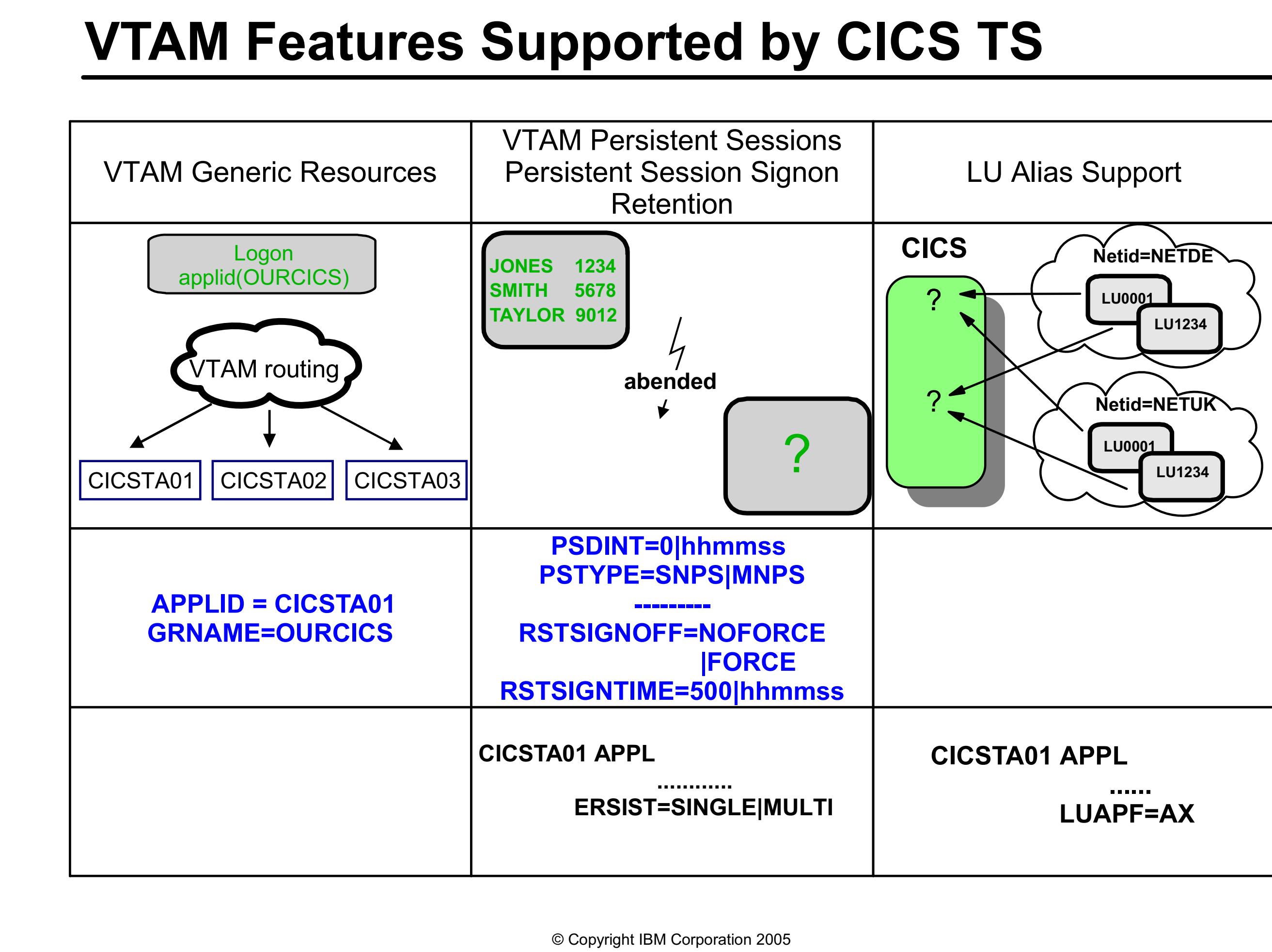


Figure 5-21. VTAM Features Supported by CICS TS CI207.0

Notes:

- **VTAM Generic Resources:** in a sysplex environment with multiple Terminal Owning Regions (TORs), Communication Server may select the region a user logon is directed to, thus balancing the workload of these TORs.
  - **GRNAME =** parameter in the SIT is set in CICS TORs to specify the “Generic Applid”. This “Generic Applid” is used by users to log on TORs.
  - z/OS sysplex with Coupling Facility is required in order to use this Communication Server feature.
- **VTAM Persistent Sessions:** if CICS fails, or is terminated through an IMMEDIATE shutdown, Communication Server may hold terminal sessions in “recovery pending” state.
  - **PSDINT=<u>0</u>|hhmmss** and **PSTYPE=<u>SNPS</u>|MNPS** are the SIT parameters to support this function. If CICS is restarted within the time specified with the PSDINT parameter, CICS restores pending sessions from the global catalog and the system

log. **M**ulti **N**ode **P**ersistent **S**ession means that the CICS region failed may be started in another z/OS image of the sysplex.

- Users are not forced off; CICS seems to “hang”. All LUs except LU 0 and LU 6.1 are supported.
- CICS may be requested to catalog user signons, and sign them on again after sessions have been rebound. Signon Retention may be determined per CICS region (SIT parameters **RSTSIGNOFF=NOFORCE|FORCE** and **RSTSIGNTIME=500|hhmmss**)), per group of terminals (Typeterm parameter **RSTSIGNOFF),** or per user(**XRFSOFF** in RACF).

- **LU Alias Support:** if terminals log on to CICS from a different network, Communication Server may present an alias LU to CICS instead of (or rather, in addition to) the terminal’s real netname(LU), thus ensuring unique names throughout a large terminal network.
  - This LU alias is treated as the terminal’s netname within CICS.
  - The NQNAME option of the INQ TERMINAL command returns the original fully-qualified network of the terminal.

SIT Parameters Summary

VTAM={<u>YES</u>|NO}	TCP={<u>YES</u>|NO}
APPLID={<u>DBDCCICS</u>|name}	
TCT={<u>YES</u>|xx|NO}	
AIEXIT={<u>DFHZATDX</u>|pgmname}	AIQMAX={<u>100</u>|number}
AILDELAY={<u>0</u>|hhmmss}	AIRDELAY={<u>700</u>|hhmmss}
AICONS={<u>YES</u>|NO|AUTO}	
GRNAME=name	
PSDINT={<u>0</u>|hhmmss}	PSTYPE={<u>SNPS</u>|MNPS}
RSTSIGNOFF={<u>NOFORCE</u>|FORCE}	RSTSIGNTIME={<u>500</u>|hhmmss}
GMTEXT={'<u>WELCOME TO CICS</u>'|'your text'}	GMTRAN={<u>CSGM</u>|txcode}
LGNMSG={<u>NO</u>|YES}	

Figure 5-22. SIT Parameters Summary CI207.0

Notes:

- Remember, SIT parameters are documented in the *CICS System Definition Guide.*
- `APPLID` tells CICS the name of the VTAM ACB it will open at startup.
- `GRNAME` specifies the VTAM generic resource name (optional usage for a TOR).
- `TCT=YES` will attempt to load a terminal control table with a blank suffix.
  "NO" should be coded for both TCP and TCT, unless you support sequential or TCAM DCB terminals.
- `GMTRAN` specifies the name of the transaction to be initiated against a terminal that has logged on to CICS through VTAM. CESN may be specified.
- Use `GMTEXT` to tailor the top line of the "Good Morning" message.
- `LGNMSG(YES)` specifies that VTAM logon data is available to application programs. Data is retrieved with an `EXEC CICS EXTRACT LOGONMSG`.

Terminal Definition (Summary)

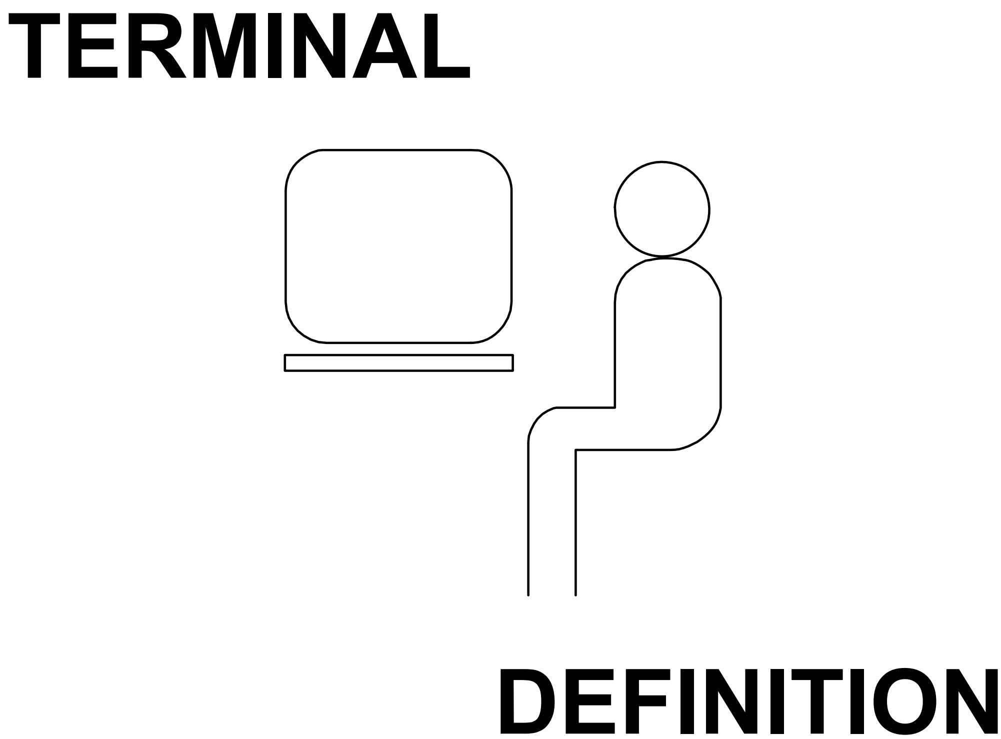


Figure 5-23. Terminal Definition (Summary) CI207.0

Notes:

- Be sure you have an understanding of the hardware and the network definitions to correctly define terminals to CICS.
- You define terminals through two related resource types: TERMINAL and TYPETERM:
  - Most of the terminal characteristics are defined in the TYPETERM.
  - The TERMINAL contains only properties unique to one device.
- The terminal autoinstall process offers savings in storage and network management, with the requirement for a user replaceable program.
- Sequential terminals must be defined with macros (TCT).
- The TYPETERM Operational Parameters specify (among others):
  - The way transactions may be initiated
  - Whether or not and how sessions may be created and terminated.
- Additional SIT parameters define the interface with VTAM.

Unit Summary

Having completed this unit, you should be able to:

- List the network information necessary to define terminals
- Explain the relationship between TERMINAL and TYPETERM definitions
- Define 3270 terminals supported by VTAM
- Explain the concepts of autoinstall
- Code SIT and VTAM parameters related to communications

Figure 5-24. Unit Summary CI207.0

Notes:

Unit 6. Defining Transactions, Programs, and Map Sets

What This Unit Is About

This unit teaches the needs and means of defining transactions the users will enter, the programs they execute, and the map sets required to format terminal data.

What You Should Be Able to Do

After completing this unit, you should be able to:

- Define transactions, programs, and map sets using RDO commands
- Explain and activate/use the program autoinstall function

How You Will Check Your Progress

- Machine lab exercise (number 4): definitions and subsequent verification

  Online definition of the invocational transactions, programs, and map sets of a real application, and the same for some lab transactions with some special requirements

References

CICS System Definition Guide, SC34-6428

CICS Resource Definition Guide, SC34-6430

Unit Objectives

After completing this unit, you should be able to:

- Define transactions, programs, and map sets using RDO commands
- Explain and activate/use the program autoinstall function

Figure 6-1. Unit Objectives CI207.0

Notes:

6.1 Defining Transactions

Basic Interdependencies

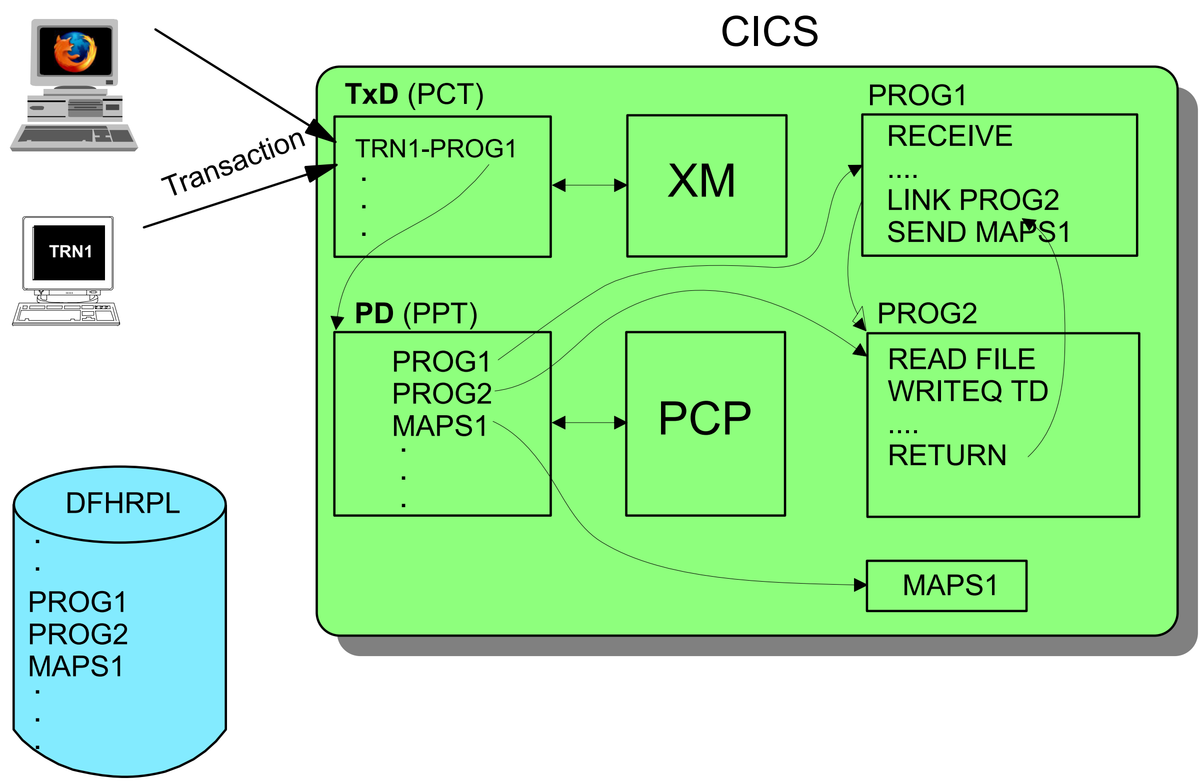


Figure 6-2. Basic Interdependencies CI207.0

Notes:

- CICS uses your transaction definitions in the CSD file to build the Transaction Directory (TxD), which is a list of valid transactions and their characteristics (the old name, Program Control Table (PCT), is still being used in some places).
  - When you define a transaction, you name the **first program** it uses.
- CICS accesses a program, loading it from the DFHRPL library concatenation into the region if it is not already in storage, whenever:
  - An operator enters a transaction
  - A program calls for program services, such as `EXEC CICS LINK`
  - A program uses `EXEC CICS SEND MAP`. In this case, the module loaded is not an executable program but a collection of structures called a **map set** which tells the CICS BMS function how to format screen input/output data.
- You may also define programs and map sets to CICS.
  Program and map set definitions are loaded into the Program Directory (PD), which is also still known as the Processing Program Table (PPT).

Defining Transactions

CEDA DEFINE TRANSACTION ...

```
TRANSaction  ==> TRN1
Group        ==> GROUPA
DEscription  ==> .....
PROGram      ==> PROG1
TWAsize      ==> 100
PROFile      ==> DFHCICST
Status       ==> Enabled
PRiority     ==> 50
```

```
PROFILE      ==> DFHCICST
Scrnsize     ==> Default
Journal      ==> NO
MSGJrnl      ==> NO
Uctran       ==> NO
RTimeout     ==> NO
```

CEDA DEFINE PROFILE ...

Figure 6-3. Defining Transactions CI207.0

Notes:

- The names of the transaction, group, and the first program or the remote attributes are mandatory.
- Each transaction is automatically associated with a **PROFILE**, which is a type of resource that defines how a transaction uses terminals, for example:
  - If the DEFAULT or ALTERNATE screen size is being used
  - Whether or not user input will be converted to uppercase
  - If EXEC CICS RECEIVE commands within conversational transactions will time out
  - Whether input or output messages, or both, shall be written to journals.

  Standard profiles are provided in the CICS-supplied group `DFHSTAND`.
- The task priority, on which CICS internal dispatching is based, is the sum of the transaction, operator, and terminal priorities.
- Remote attributes and security will be explained in other units.

6.2 Defining and Handling Programs and Map Sets

Define Program

```
PROGram              :  PROG1
Group                :  GROUPA
DEscription      ==>  Customer inquiry
Language         ==>  Cobol|Assembler|Le370|C|Pli
RELoad           ==>  No | Yes
RESident         ==>  No | Yes
USAge            ==>  Normal | Transient
Status           ==>  Enabled | Disabled
COncurrency      ==>  Quasirent | Threadsafe
Api              ==>  Cicsapi | Openapi
......           ......
JVM ATTRIBUTES
 JVM             ==>  No |Yes
 JVMClass        ==>  example.HelloWorld.CICS
```


Figure 6-4. Define Program CI207.0

Notes:

- Program definition is not mandatory, but you still may have special needs for some of your programs that require a program definition.

RELoal(YES)	A fresh copy of the program is brought into storage for every request. Furthermore, each of these program copies must be removed from storage explicitly.
RESident(YES)	The program will be loaded on first reference, and then stay permanently in virtual storage, even if not in use.
USAge(TRANSIENT)	Storage for this program is immediately released when the program is not in use.
Concurrency(Threadsafe)	The program is written to threadsafe standards, and may run entirely or partially on L8 or L9 TCB.
Api(Openapi)	The program is not restricted to the CICS application program interfaces.
JVM(Yes)	The program is to operate under a JVM. Specify a class name in the JVMCLASS attribute if you specify JVM(YES).

Define Map Set

Mapset	:	MAPS1
Group	:	GROUPA
DEscription	==>	Inquiry Dialog Mapset
RESident	==>	No | Yes
USAge	==>	Normal | Transient
Status	==>	Enabled | Disabled

Figure 6-5. Define Map Set CI207.0

Notes:

Each 3270 interactive application using a display device can use specific screen layouts, or *maps*. These are not specified in the program itself. Instead, you use basic mapping support (BMS). This gives greater flexibility and allows the maps to be used by multiple invocations of the same program, or by several different programs.
You specify maps, and the fields on them, using CICS-supplied macros, and assemble and link-edit sets of related maps as map set load modules.

- CICS manages map sets much like programs, in that they are loaded from DFHRPL when an application issues a SEND/RECEIVE MAP command.
- As for programs, map set names can be up to 8 characters in length.
- The map set definition set is a subset of the program definition.

Inquiring Programs and Map Sets

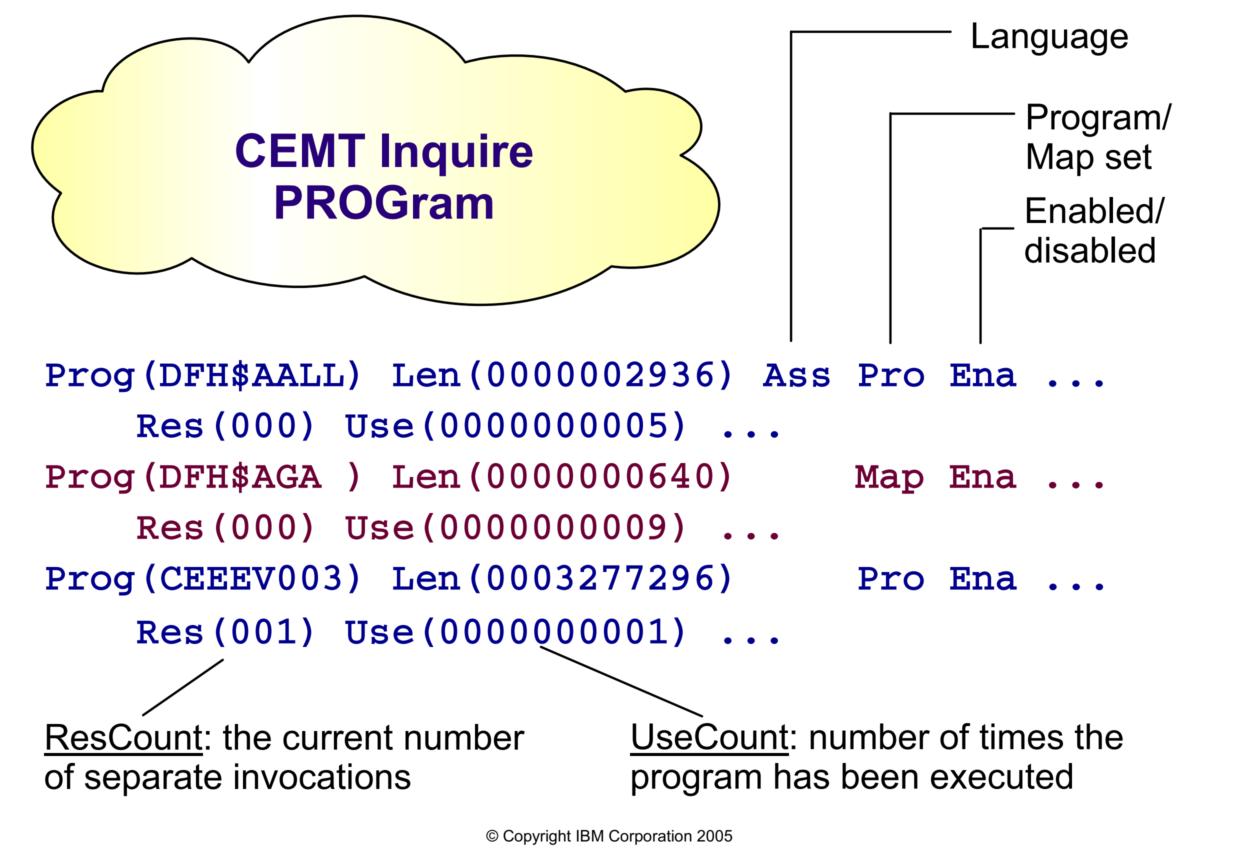


Figure 6-6. Inquiring Programs and Map Sets CI207.0

Notes:

Parts of the information stored in the Program Directory main storage table are displayed as a result of the CEMT INQ PROG command.

- This command serves both program and map set entries. Programs are marked “Pro”, while map sets are marked “Map” in the appropriate column.
- Notice two counter fields displayed, which explain about the current and the overall use of each program within this CICS region:
  - The RESCOUNT value identifies the number of separate invocations of this program that are taking place at the time of this inquiry.
  - The USECOUNT value identifies the total number of times the program has been executed since the start of the current CICS session.

Loading a New Copy

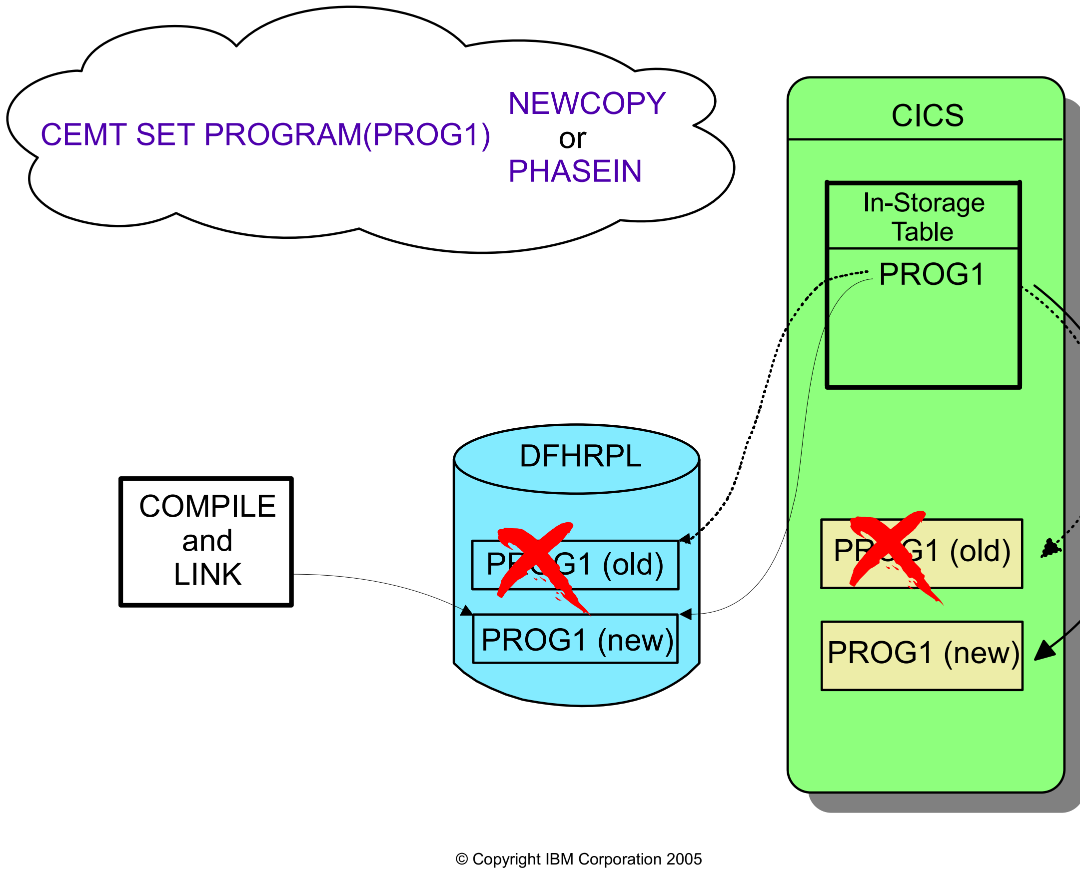


Figure 6-7. Loading a New Copy CI207.0

Notes:

- In addition to what you specify when you define resources, CICS uses tables like the program directory (PPT) to maintain information like the DASD address of the program.
- After you recompile and link PROG1, the program directory still points to the old version.
- The master terminal (CEMT) transaction's **PHASEIN** option causes CICS to refresh the pointer. CICS is to use a new copy of the program now for all new transaction requests. CICS continues to use the old copy for all currently running transactions until they have finished.
- Another command, **NEWCOPY**, also tells CICS to use a new copy of the program, but it cannot be performed unless the program is not in use.
- **PHASEIN** and **NEWCOPY** cannot be specified for a JVM program.

Do you also need to repeat the CEDA INSTALL command after linking a new version of the program?

6.3 Program Autoinstall

Program Autoinstall

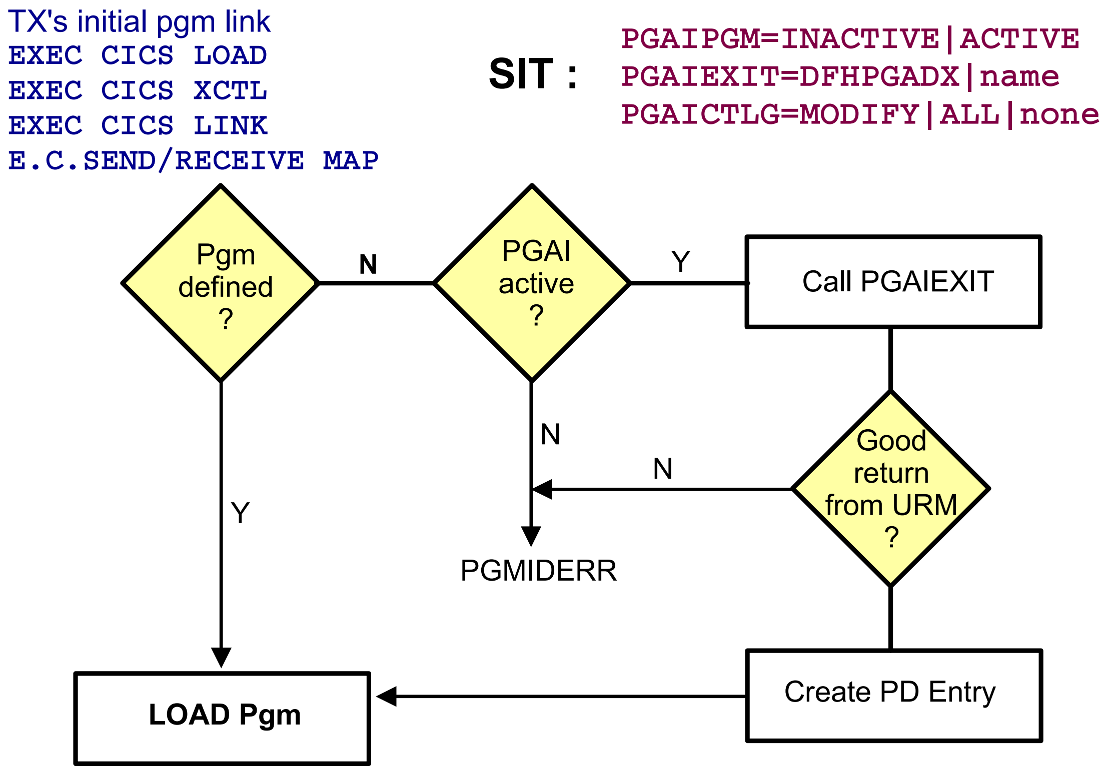


Figure 6-8. Program Autoinstall CI207.0

Notes:

- A user-replaceable module (URM) may determine:
  - A valid program name that is taken as a model or template for the program definition to be dynamically created

  and/or:

  - Single program definition parameters to be activated
- **Input to URM is:**
  - Name of the program or map set
  - Module type (Program | Map set | Partitionset)
- The default exit program **DFHPGADX** is contained in installation libraries `SDFHLOAD` (executable module) and `SDFHSAMP` (Assembler source).
- Default models **DFHPGAPG** (for programs) and **DFHPGAMP** (for map sets) are contained in IBM-supplied group **DFHPGAIP**.

Program Load Status

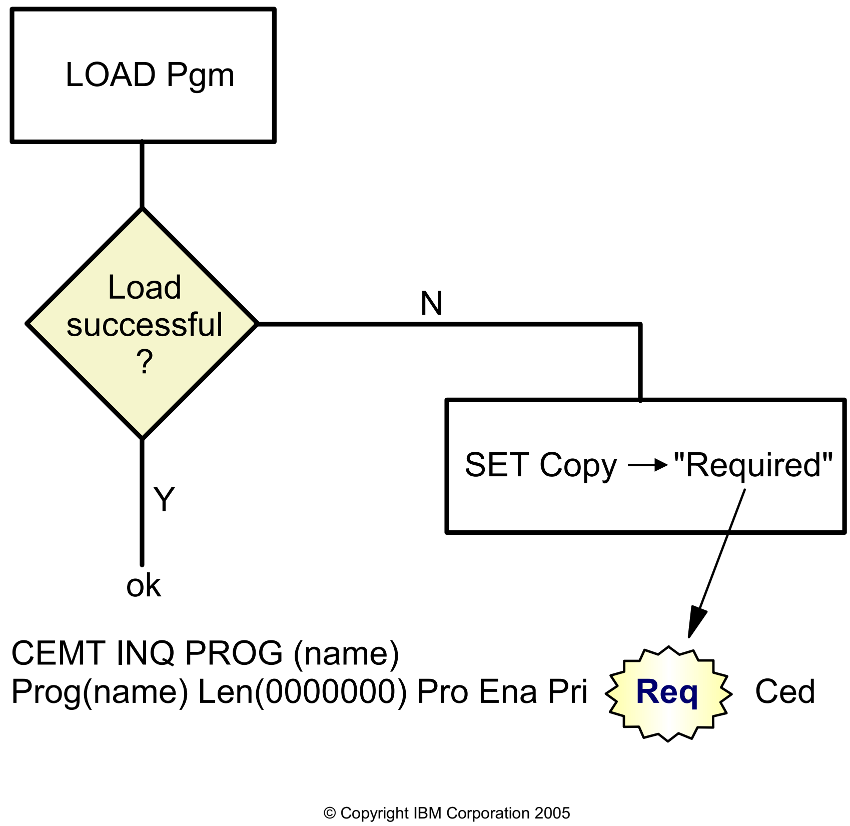


Figure 6-9. Program Load Status CI207.0

Notes:

- Programs — whether or not autoinstalled — that cannot be found in DFHRPL will be marked in the Program Directory (PD/PPT).
- On subsequent calls of the program (LOAD, LINK, XCTL), no load will be done, but the calling program will be returned the `PGMIDERR` condition.
- The `NOT_LOADABLE / (Copy) REQUIRED` status can only be resolved by a `SET PROG(...) NEWCOPY command`.
- Just to refer the additional status information shown in the visual:
  - Pri Indicates that the program is "private", which means that it has been, or will be, loaded from the DFHRPL concatenation, as opposed to being run from the z/OS LPA.
  - Ced Indicates that the program has been prepared to support the Execution Diagnostic Facility (CEDF, CEDX transactions).

Transactions, Programs, Map Sets (Summary)

Defining Transactions, Programs, and Map Sets

Figure 6-10. Transactions, Programs, Map Sets (Summary) CI207.0

Notes:

- Transactions that are to be processed by CICS TS have to be defined using a unique (up to) 4-character transaction code.
- Only the name of the initial program of a transaction has to be defined within the transaction definition.
- Within a transaction, several programs may be invoked by means of the commands EXEC CICS LINK, EXEC CICS XCTL, and/or EXEC CICS LOAD.
- Programs and map sets may be autoinstalled based on a model definition that is selected by a user-replaceable module.
- Usually, programs and map sets have to be loadable from a load library contained in the DFHRPL concatenation of the CICS TS startup JCL.

Unit Summary

Having completed this unit, you should be able to:

- Define transactions, programs, and map sets using RDO commands
- Explain and activate/use the program autoinstall function

Figure 6-11. Unit Summary CI207.0

Notes:

Unit 7. File Control Functions and File Definition

What This Unit Is About

This unit describes the CICS TS File Management facilities and the resource definition types and parameters that address these.

What You Should Be Able to Do

After completing this unit, you should be able to:

- List the file organizations supported by CICS File Management
- State the use of VSAM strings by File Management and their definition within CICS
- Describe VSAM shared resources and state the benefits
- Define VSAM files to support user applications
- Use SIT parameters to define DFHCSD
- Identify the requirements for a VSAM Alternate Index
- Describe the Shared Data Tables function and state its benefits

How You Will Check Your Progress

- Machine lab exercise 5:
  Online definition of file resources with subsequent online verification

References

CICS System Definition Guide, SC34-6428

CICS Resource Definition Guide, SC34-6430

Unit Objectives

After completing this unit, you should be able to:

- List the file organizations supported by CICS File Management
- State the use of VSAM strings by File Management and their definition within CICS
- Describe VSAM shared resources and state the benefits
- Define VSAM files to support user applications
- Use SIT parameters to define DFHCSD
- Identify the requirements for a VSAM Alternate Index
- Describe the Shared Data Tables function and state its benefits

Figure 7-1. Unit Objectives CI207.0

Notes:

7.1 File Definition Basics

File Management

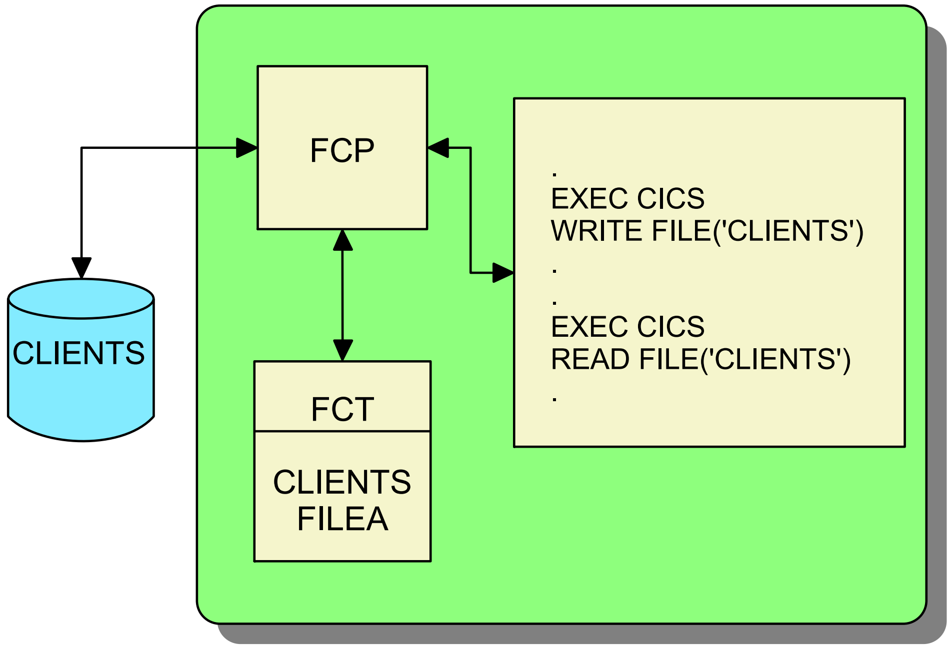


Figure 7-2. File Management CI207.0

Notes:

- All requests for file services are passed to the File Control Program (FCP).
- All online VSAM and BDAM data sets accessed through File Control must be defined to CICS.
- VSAM data set attributes are contained in the VSAM catalog. Information necessary for CICS to access the file is included in your file definition.

CICS Support for VSAM Facilities

KSDS	Access by key
ESDS	Access by entry number
RRDS	Access by record address
AIX	Alternate Index
NSR	Resources dedicated per file
LSR	Resources shared by a group of files
RLS	Shared access by multiple CICS regions in a sysplex

Figure 7-3. CICS Support for VSAM Facilities CI207.0

Notes:

- CICS File Management supports VSAM:
  - Key-sequenced data sets (KSDS)
  - Entry-sequenced data sets (ESDS)
  - Relative Record data sets (RRDS)
  - Alternate Indexes (AIX)
  - Non Shared Resources (NSR)
  - Local Shared Resources (LSR)
  - Record Level Sharing (RLS)
- Any VSAM facilities must be defined to RDO.
- BDAM data sets are also supported, but can only be defined with macros. The *CICS Resource Definition Guide, SC34-6430*, contains documentation for macro file definition.

Static Allocation

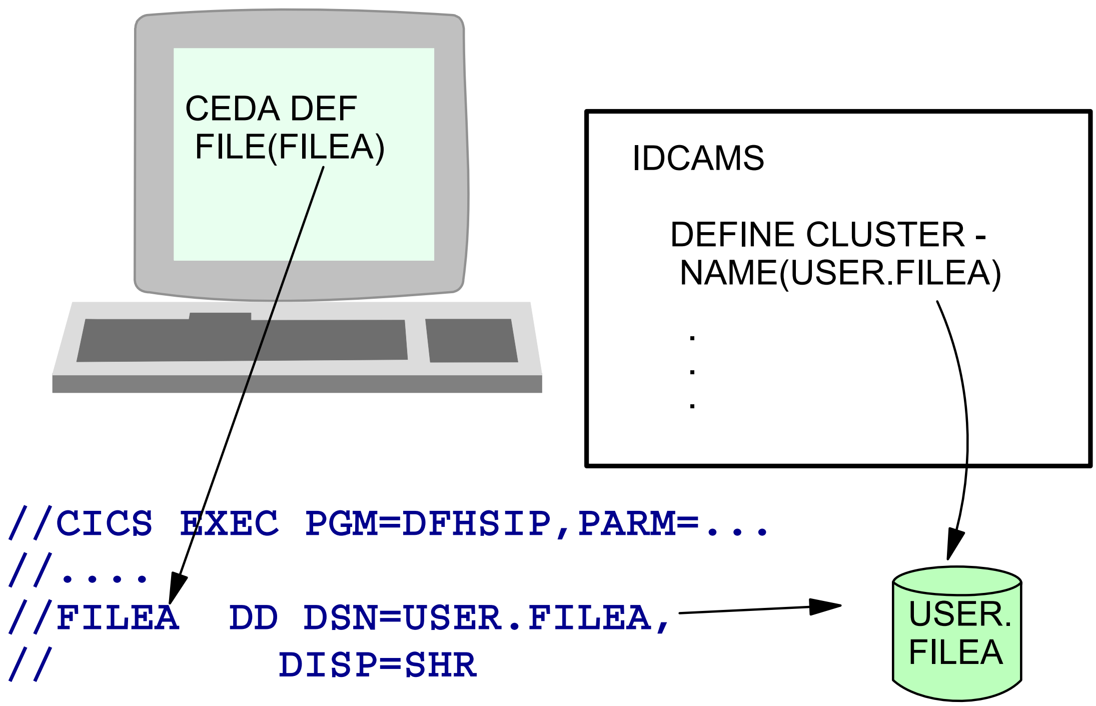


Figure 7-4. Static Allocation CI207.0

Notes:

- CICS knows a file by its DDname.
  Application programs have to use this name within file requests.
- The Access Method Services utility defines a VSAM cluster by its Data Set Name (DSN).
- Supplying a DD Statement in the CICS startup job/procedure is one way to identify the data set to CICS. Doing this, however, allocates the file when CICS starts up, and deallocates it when CICS terminates.
- Coding DD statements in CICS JCL can cause a problem if:
  - You need to run a batch job accessing a CICS file
  - The file does not exist when CICS is initialized
  - You want to back up or recover a file.

Dynamic Allocation

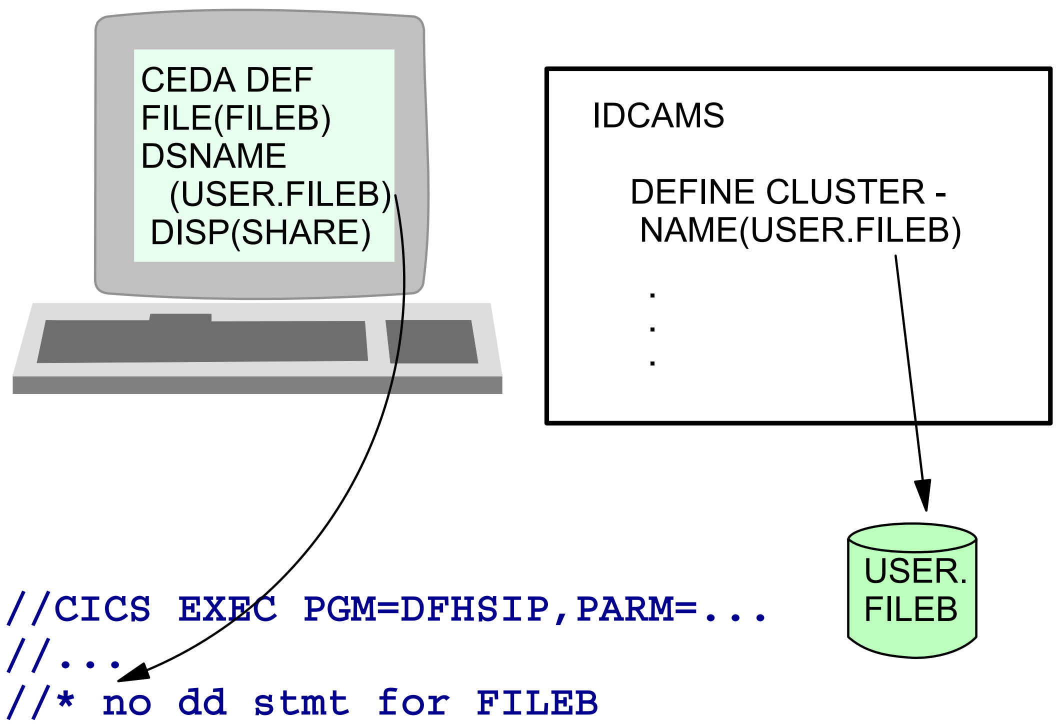


Figure 7-5. Dynamic Allocation CI207.0

Notes:

- Omit DD statement for data set
- Specify DSNAME and DISPOSITION in file definition

  This results in dynamic allocation and deallocation when opening or closing of the file is requested, either explicitly by commands, or implicitly by application requests.
- For closed files, DSNAME and DISPOSITION may be changed while CICS is executing, using `EXEC CICS/CEMT SET FILE` commands.

What would happen at open time if you omitted the DD statement and did not have DSNAME and DISPOSITION in your file definition?

What if you included both the DD statement and the file definition parameters for data set name and disposition?

Initial File States

CEDA DEF FILE(filename) GROUP(rdogrp)

Opentime

STAtus()	Startup	Firstref
Enabled	Opened by CSFU Right after Initialization	● Explicit Open by command ● Implicit Open by application request
Disabled		● Explicit Open satisfied ● DISABLED except.condition on application request
Unenabled	N/A	● Explicit Open satisfied Status→Enabled ● NOTOPEN except.condition on application request

Figure 7-6. Initial File States CI207.0

Notes:

The combination of STATUS and OPENTIME can impact performance and application program exception condition handling.

- You should define files as `OPENTIME(FIRSTREF)` and `STATUS(ENABLED)`. These files are opened when first referenced. This is the default setting.
- Files defined as `OPENTIME(STARTUP)` are opened by a CICS transaction after control is given to CICS. This causes extra overhead, and should not be used unless files need to be opened at that time.
- Define files as `OPENTIME(FIRSTREF)` and `STATUS(UNENABLED)` if you want them to remain closed until opened by an operator or application program request.
- Closing a file causes `OPENTIME(FIRSTREF)` and `STATUS(UNENABLED)` to be set.

Changing File States

Master Terminal

CEMT INQ FILE

CEMT SET FILE CLOSED

OPEN | CLOSED |
FORCECLOSE

ENABLED |DISABLED

Application
Program

```
        .
        .
EXEC CICS SET
  FILE ('FILEA') CLOSED
  BUSY (FORCE|WAIT|NOWAIT)
        .
        .
        .
```


Figure 7-7. Changing File States CI207.0

Notes:

What will CICS do if the file is in use when a `SET CLOSE` or `SET DISABLED` command is issued with the busy option?

FORCE Abends all tasks that use the file, and performs the request immediately (handle with care!).

WAIT The request is queued until all activity against the file has quiesced. Control will not be returned to the issuing application before the SET has been performed.

NOWAIT Same as WAIT, but control is returned to the issuing application as soon as the request has been queued.

DEFINE/SET Operational Options

```
CEDA  DEFine File( FILEX    )

 OPERATIONS
  Add             ==> No
  BRowse          ==> No
  DELete          ==> No
  READ            ==> Yes
  UPDATE          ==> No
```

issue command, such as:
CEMT SET FI(..) UPDATE
or change here

```
CEMT i file
STATUS:  RESULTS - OVERTYPE TO MODIFY
 Fil(CI20FCAI) Vsa Ope Ena Rea         Bro         Sha
        Dsn( DECI200.AIPTH                                   )
 Fil(CI20FCUS) Vsa Ope Ena Rea Upd Add Bro Del     Sha
        Dsn( DECI200.CTCUS                                   )
 Fil(DFHCMACD) Vsa Clo Ena Rea                     Sha
        Dsn( DECI200.DFHCMACD                                )
 Fil(DFHCSD  ) Vsa Clo Une Rea Upd Add Bro Del     Sha
        Dsn( DECI200.DFHCSD                                  )
```


Figure 7-8. DEFINE/SET Operational Options CI207.0

Notes:

- You can specify which application requests will be processed for a specific file.
  - A denied application request will return the `INVREQ` exceptional condition to the application.
- Operational options may be changed dynamically using CEMT, or by application programs using the EXEC CICS SET FILE command.
- Files must be closed before an option can be changed.
- If only READ and/or BROWSE is permitted, CICS will open the file with the READONLY options; otherwise with READWRITE.

7.2 Sharing Resources

VSAM Strings

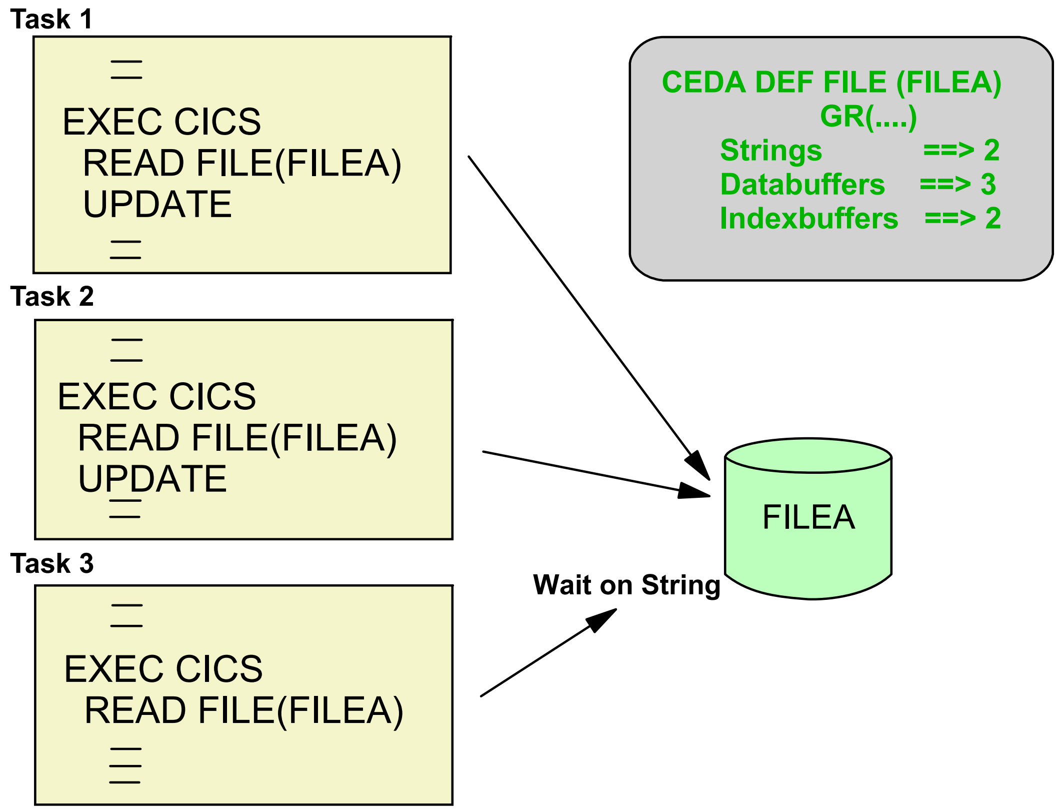


Figure 7-9. VSAM Strings CI207.0

Notes:

In order to access a VSAM file, you need two things: one string (or placeholder) and one or more buffers.

- You may specify the number of strings and buffers within the file definition.
  - `INDEXBUFFERS(n)` specifies the number of buffers for index CIs; the default is one per string.
  - `DATABUFFERS(n)` defines the number of buffers reserved for data CIs; the default is one buffer per string plus one.

  To optimize specifications, you have to know the number of the files' index levels and to increase the number of buffers for index:

  - As a minimum, specify (number of strings) + (index levels -1). This value ensures some *lookaside* at the high index level. Lookaside is the process where VSAM locates the requested data in the buffer, eliminating the need to do physical I/O.
  - As a maximum, specify the number of CIs in index set + 1. With this value, all index set buffers will remain in storage.

- Strings are held for varying lengths of time, depending on the type of request:
  - **READ, WRITE, DELETE** During request precessing
  - **UPDATE, DELETE** READ UPDATE until REWRITE or DELETE
  - **BROWSE** STARTBR until ENDBR
  - **MASS RECORD INSERT** WRITE (Massinsert) to UNLOCK

Non or Local Shared Resources (NSR or LSR)

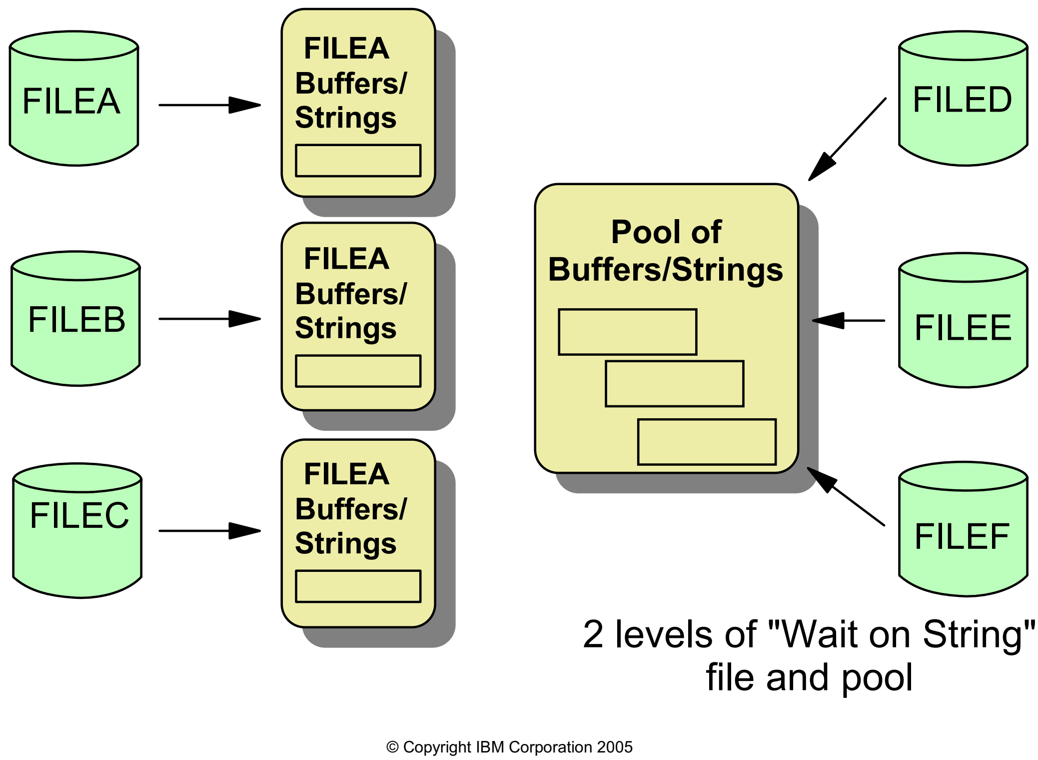


Figure 7-10. Non or Local Shared Resources (NSR or LSR) CI207.0

Notes:

When you allocate dedicated strings and buffers to each file (**N**on **S**hared **R**esources), if such a file needs additional resources, CICS cannot borrow them from another file. The solution is a common pool of strings and buffers associated with a group of files; this is the basic concept of **L**ocal **S**hared **R**esources.

- CICS file control uses up to eight different LSP pools. During file definition, you determine which pool a file use; the default value is pool number one.
- You should define each pool to best satisfy your installation’s requirements.
- LSR should reduce the amount of I/Os through “lookaside”.
- FCP first checks to see if a file string is available. If not, the task waits for a file string. Then, FCP checks for an available pool string. If not, the task waits for a pool string.

LSR Pool Definition

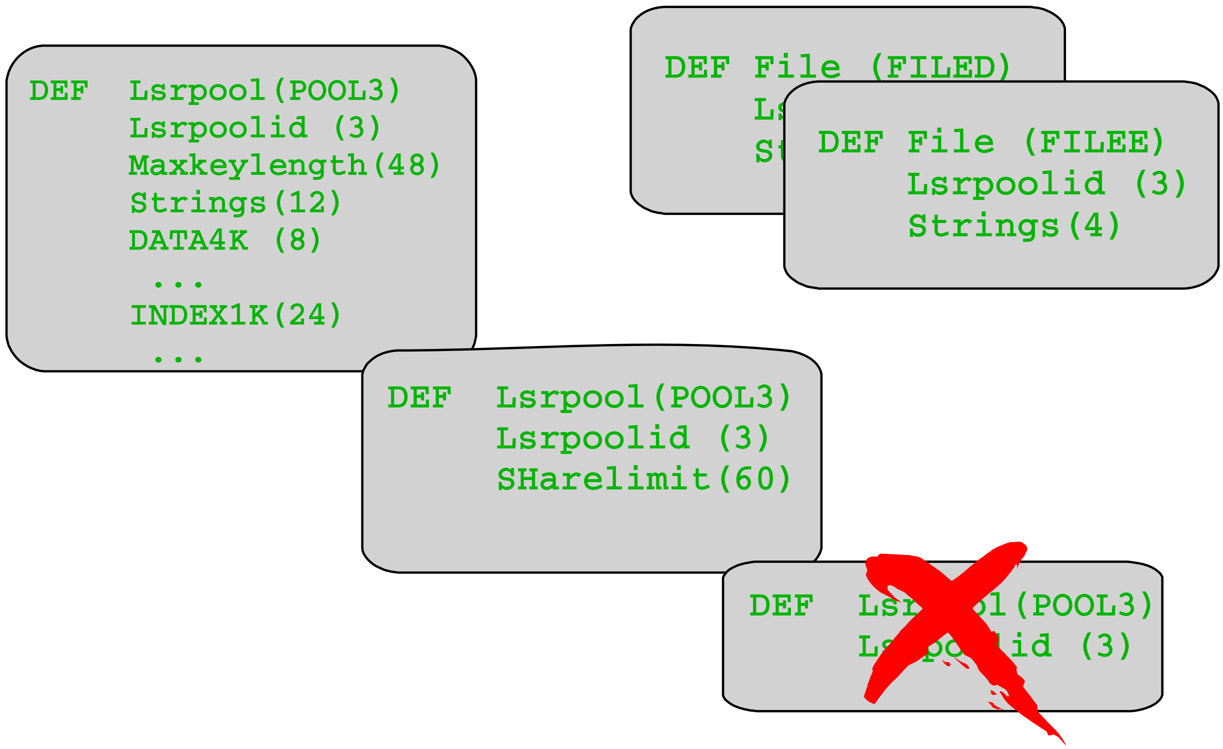


Figure 7-11. LSR Pool Definition CI207.0

Notes:

You should use local shared resources where possible.

- If you do **not define an LSR pool**, CICS calculates buffers and strings using your string specification for all files in the pool and information in the VSAM catalog. CICS multiplies its calculated buffer and string requirement by SHARELIMIT. The default is 50%. CICS always allocates a minimum of three buffers of each size. The *CICS Performance Guide, SC34-6452*, describes how CICS calculates the default pool size.
- You may **define an LSR pool** using only the `SHARELIMIT` parameter, and let CICS calculate the rest.
- If you want to control pool creation and eliminate the need to access the VSAM catalog, you have to **provide a complete LSR pool definition**:
  - *maxkeylength*: Must accommodate the longest key of any file in the pool.
  - *strings*: Specifies the limit of all the strings of the files in the pool. The string numbers specified in the file definition ensure that access to one file cannot occupy the whole pool of strings.

- *buffers*: Specifies the number of data or index buffers of each size that you require, in the range 3 through 32767. If you leave this field blank, there are no default values. You must define buffers large enough to contain the largest CI size.

VSAM Alternate Index

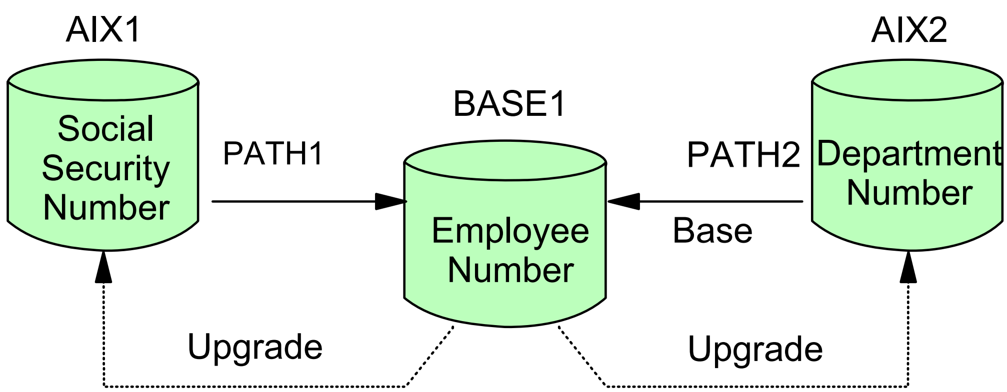


Figure 7-12. VSAM Alternate Index CI207.0

Notes:

- The IDCAMS definition for an alternate index (AIX) consists of three components:
  a. BASE cluster, containing the user data.
  b. AIX cluster, defining an alternate access to the base.
  c. PATH entry, defining the relationship between the AIX and BASE clusters.
- The UPGRADE parameter tells VSAM to update the alternate indexes when the base AIX keys are changed.
- If your applications need access to the base cluster, include the base cluster in your file definitions.
- To access the base through the alternate index, **define the paths** (PATH1 and PATH2) as a file, not the alternate indexes (AIX1, AIX2).

CSD File Definition

- CSDDSN = dsname
- CSDDISP = OLD | SHR
- CSDACC = <u>READWRITE</u> | READONLY
- CSDLSRNO = <u>1</u> | nbr | NO/NONE
- CSDSTRNO = number
- CSDBUFND = number
- CSDBUFNI = number

Figure 7-13. CSD File Definition CI207.0

Notes:

- Define DFHCSD using the SIT, as DFHCSD cannot be defined within itself.
- CSD definition parameters will be effective only on a CICS cold or initial start.
- As with other VSAM files, DFHCSD may be allocated statically or dynamically, depending on where its data set name and disposition is provided, either as a DD statement or a SIT parameter.
  Note that there is no default for `CSDDISP`. If you specify the DFHCSD data set name via the `CSDDSN` parameter, you must also specify an option for `CSDDISP`.
- If you plan to share DFHCSD, only one CICS should have update access (`CSDACC=READWRITE`). All other CICS regions using that CSD file should have `CSDACC=READONLY`. This allows multiple address spaces to have the CSD file open at the same time with VSAM SHR(2,3).
- DFHCSD may be run in either NSR or LSR mode, just as any user data file.
- CICS requires two strings per CSD user. To allow more than one concurrent CEDA user, increase the CSDSTRNO value, in multiples of two.

7.3 Shared Data Tables

Shared Data Tables Overview

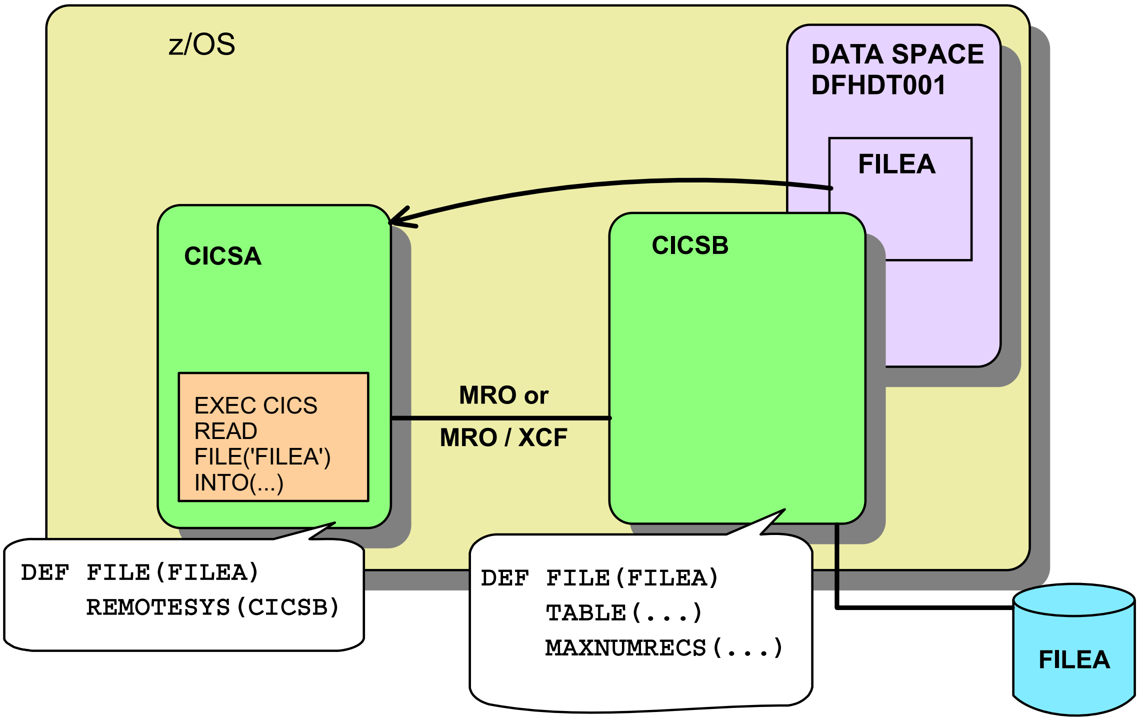


Figure 7-14. Shared Data Tables Overview CI207.0

Notes:

- A data table is essentially an in-storage copy of a VSAM file or parts of it.
- You define data tables as files. The TABLE value identifies this file as a data table, and the MAXNUMRECS defines the table size.
- When the file is opened, the data table is automatically created in an z/OS *data space* named **DFHDT001**, owned by the file owning CICS region.
- Read-only requests against a file that is owned by a remote CICS within the same z/OS image are satisfied directly from the data space by the SDT feature, without running the function shipping facilities. This substantially improves read performance!
- Multiple files may be loaded into the same data space.
- The data space is created at the first OPEN, and freed at the last CLOSE of a DT file.
- A data table index and control information is held in the DSA above 16 MB.
- Only the file owning region specifies the data table in its file definition.

CICS Maintained Table (CMT)

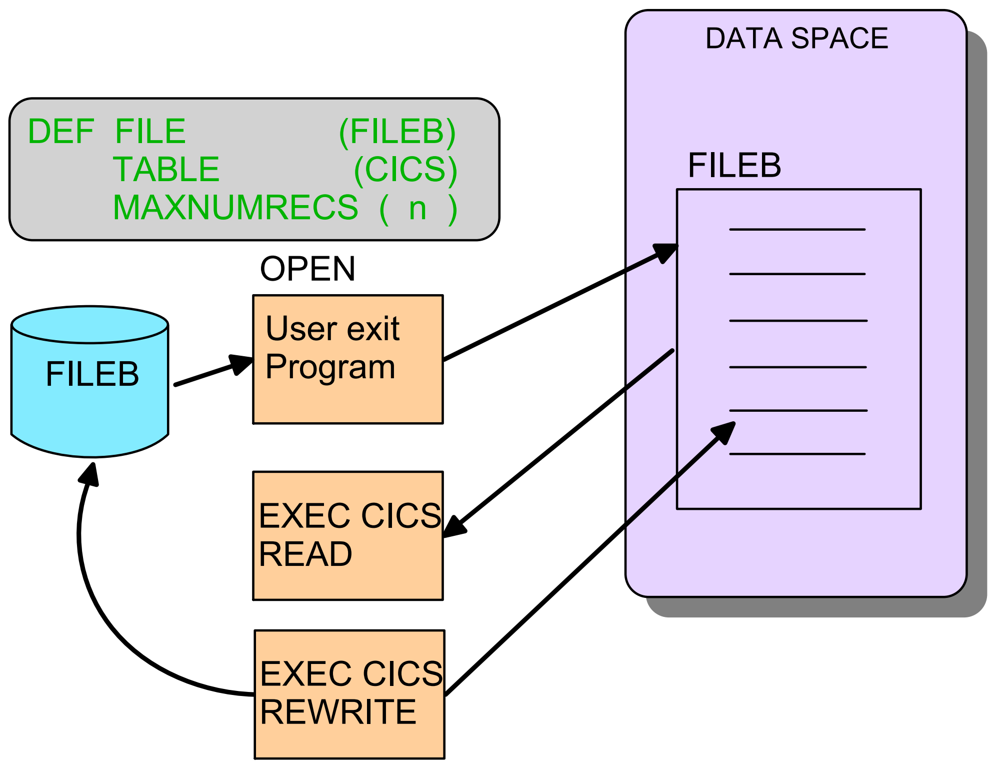


Figure 7-15. CICS Maintained Table (CMT) CI207.0

Notes:

- All records in the opened file are loaded into the table. You can write a user exit to perform record selection at load time.
- CMTs are transparent to the application; all file control commands are supported.
- Any read-only operations (READ and BROWSE) are serviced from the table without referencing the source data set.
- WRITE requests (ADD, DELETE, REWRITE) update both the table and source data set, thus providing full synchronization between the data table and the source data set.
- Recovery services for CICS-maintained tables are the same as for the source VSAM data set.

User Maintained Table (UMT)

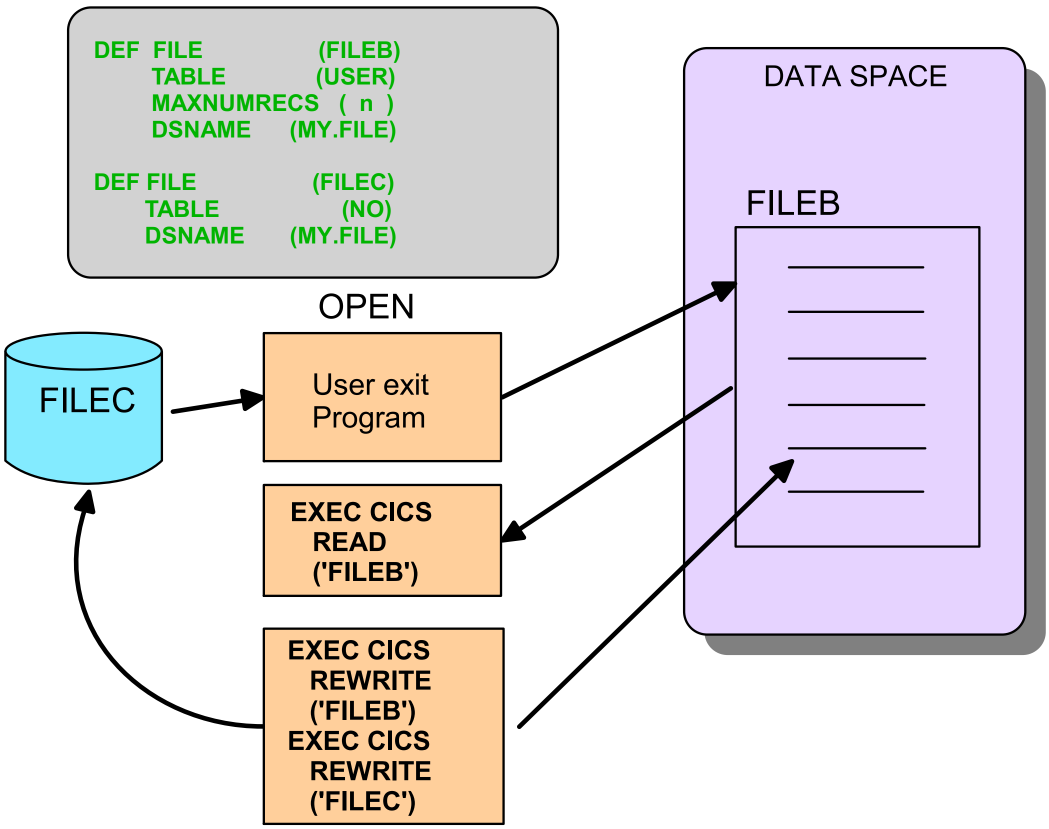


Figure 7-16. User Maintained Table (UMT) CI207.0

Notes:

User-maintained data tables require a bit more user effort, but allow more flexibility.

- As with CMT, the table is created when the file is opened for the first time.
  A user exit may select the records to be loaded, and may change the record layout to better suit your requirements; the exits for both CICS and user-maintained tables are documented in the *CICS Customization Guide, SC34-6429*.
- Only a subset of the file control API commands is supported.
- The source file is closed after the table is created, and any requests are performed only against the table.
- The user is responsible for updating any source files used to create the table.
  This requires a separate file definition (FILEC, shown here).
- If your data table is created from sources other than a VSAM KSDS, loading occurs when an application WRITEs to the table.
  A “dummy” VSAM KSDS that defines the key position and length is required.
- Recovery of a UMT is limited to dynamic transaction backout.

File Definition (Summary)

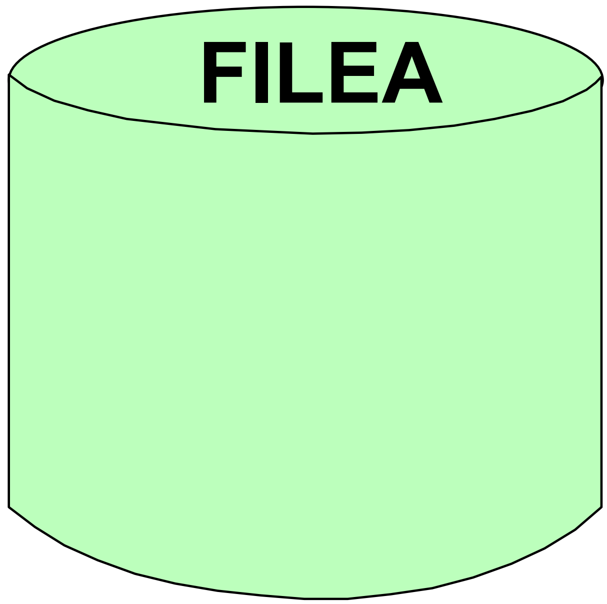


Figure 7-17. File Definition (Summary) CI207.0

Notes:

- CICS file control supports VSAM and BDAM access methods.
- Each data set must be defined to CICS:
  - VSAM files have to be defined with RDO.
  - BDAM files can only be defined with DFHFCT macro statements.
  - The DFHCSD file characteristics are defined in the SIT.
- With dynamic allocation, files are allocated to CICS only while they are open.
- Local Shared Resources (LSR) allow VSAM to pool the buffers and strings for multiple data sets. With LSR pools, the capability for “lookaside” for any buffer in the data set should decrease I/O requirements.
- You define the path to an alternate index as a CICS file.
- Shared data tables are an efficient way to process files with a high read-to-write ratio in an MRO environment.

Unit Summary

Having completed this unit, you should be able to:

- List the file organizations supported by CICS File Management
- State the use of VSAM strings by File Management and their definition within CICS
- Describe VSAM shared resources and state the benefits
- Define VSAM files to support user applications
- Use SIT parameters to define DFHCSD
- Identify the requirements for a VSAM Alternate Index
- Describe the Shared Data Tables function and state its benefits

Figure 7-18. Unit Summary CI207.0

Notes:

Unit 8. CICS TS Queuing Facilities

What This Unit Is About

This unit introduces the queuing facilities of CICS TS — Transient Data and Temporary Storage Queue Management — and shows you how to define queues for CICS and application use.

What You Should Be Able to Do

After completing this unit, you should be able to:

- Describe the CICS Transient Data and Temporary Storage queuing facilities
- Contrast intrapartition and extrapartition queues
- Describe Automatic Transaction Initiation (ATI)
- Define transient data queues using RDO to support applications using intrapartition, extrapartition, and indirect queues, and ATI
- Contrast main and auxiliary Temporary Storage
- List CICS uses of queuing facilities
- Name the CICS system data sets used for queues
- Specify the required SIT parameters to support queuing

How You Will Check Your Progress

- Machine lab exercise 6: definition and subsequent verification

  Several transient data queues are to be defined or redefined that will be used by a given lab application.

References

CICS System Definition Guide, SC34-6428

CICS Resource Definition Guide, SC34-6430

Unit Objectives

After completing this unit, you should be able to:

- Describe the CICS Transient Data and Temporary Storage queuing facilities
- Contrast intrapartition and extrapartition queues
- Describe Automatic Transaction Initiation (ATI)
- Define transient data queues using RDO to support applications using intrapartition, extrapartition, and indirect queues, and ATI
- Contrast main and auxiliary Temporary Storage
- List CICS uses of queuing facilities
- Name the CICS system data sets used for queues
- Specify the required SIT parameters to support queuing

Figure 8-1. Unit Objectives CI207.0

Notes:

8.1 Transient Data Queues

Queuing: What For?

Facility to store information
for late retrieval and use.

Two facilities in CICS:

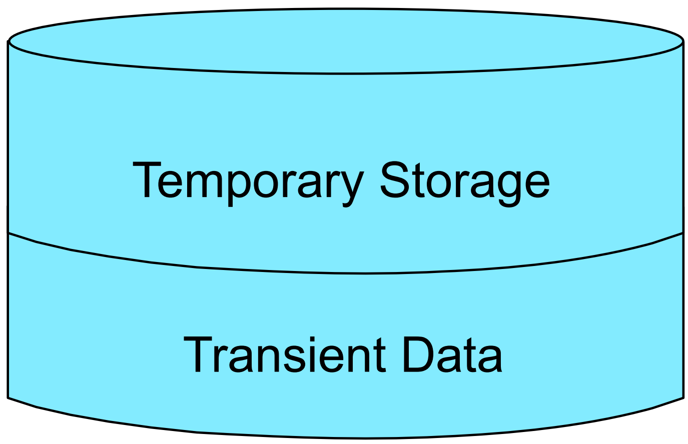


Figure 8-2. Queuing: What For? CI207.0

Notes:

- Online applications often require the ability to store data for later retrieval by another CICS task or a batch job.
- CICS supports two facilities for queuing data online:
  - Transient Data (TD)
  - Temporary Storage (TS)

Types of Transient Data

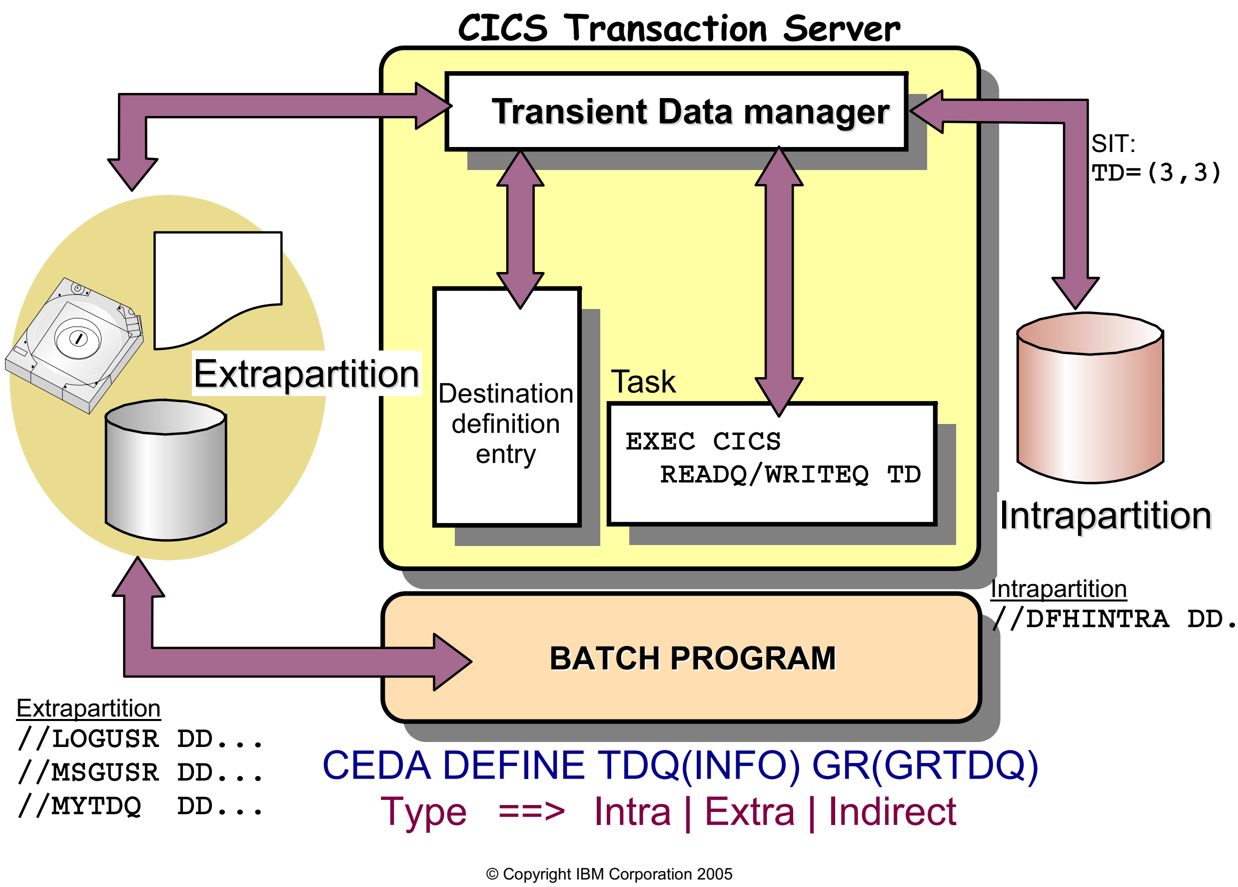


Figure 8-3. Types of Transient Data CI207.0

Notes:

TD queues must be predefined; queue names are one to four characters long, and are often referred to as “destination IDs”.

- Queues may be defined as INTRApartition, EXTRApartition, INDIRECT, or REMOTE.
  - Only CICS programs may write and read **INTRA TD queues**. All INTRA queues share a common VSAM entry sequenced data set named DFHINTRA.
  - Either CICS or batch programs may process **EXTRA TD queues**, which are MVS sequential data sets.
  - **Indirect queues** route data to INTRA, EXTRA, or other indirect queues. They contain no data themselves.
  - **Remote Queue** definitions permit access to transient data queues that are owned by other CICS regions through function shipping.
- Records are always written and read sequentially.

Sharing the Intrapartition Data Set

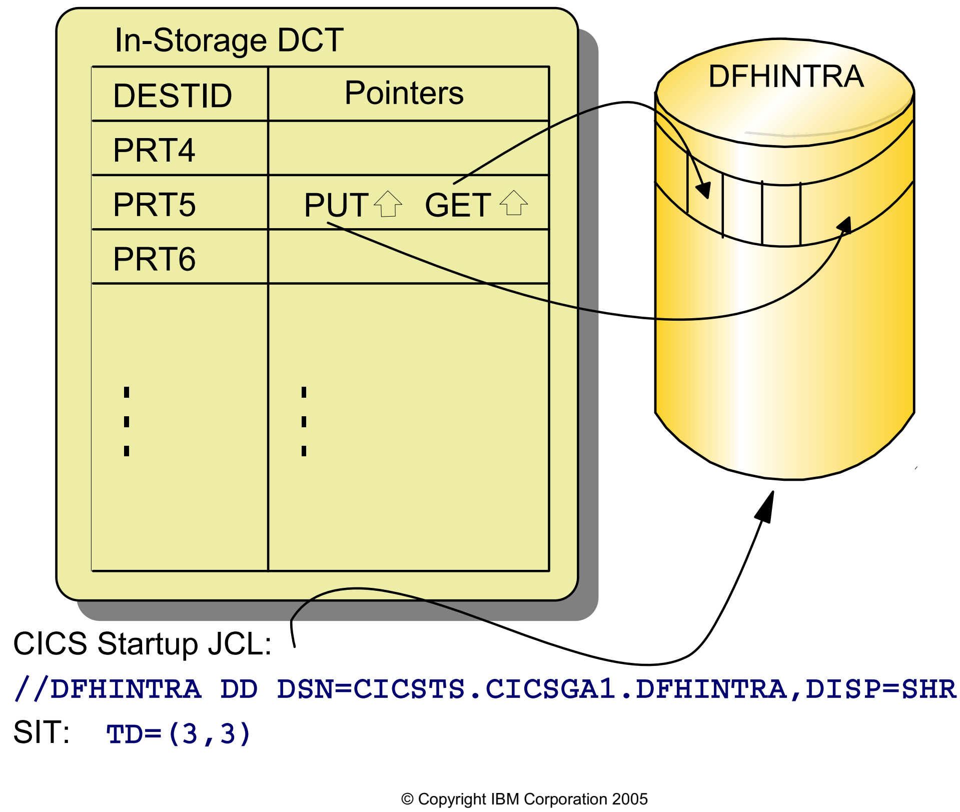


Figure 8-4. Sharing the Intrapartition Data Set CI207.0

Notes:

- Each INTRA partition queue is allocated from a single VSAM entry sequenced data set named DFHINTRA. It is one of the CICS data sets that may not be shared between regions, and it must be statically allocated by the appropriate DD statement.
- Each WRITEQ places a record in the next sequential location in the queue.
- Each READQ retrieves the current record and positions the queue pointer to the next sequential record. After a record has been read from a queue, it may not be read again. This is called a “destructive read” or a non-repeatable read.
- The CONTROLINTERVALSIZE parameter must be large enough to hold the longest record, plus the 32 bytes that CICS requires for its own purposes.
- The SIT parameter TD=(b,s) specifies the number of VSAM buffers and strings available for intrapartition TD. The default is (3,3); up to 32,767 buffers may be specified.
- The “Setting Up Transient Data Destination Data Sets” chapter of the *CICS System Definition Guide, SC34-6428*, contains information for DFHINTRA allocation.

Automatic Transaction Initiation (ATI)

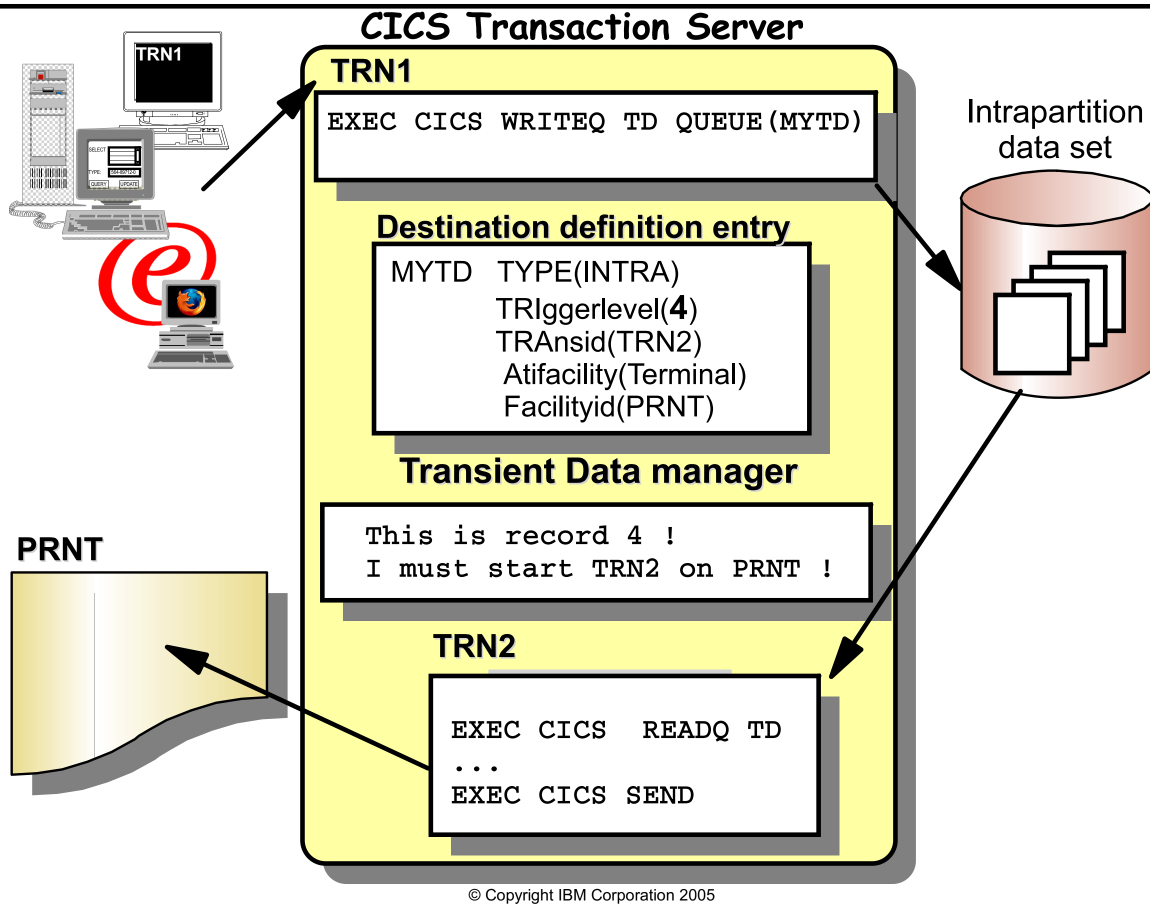


Figure 8-5. Automatic Transaction Initiation (ATI) CI207.0

Notes:

CICS architecture allows a task to use no more than one principal facility, which is normally a terminal. What if you want to run a CICS transaction on an unattended terminal, like a printer?

Intrapartition TD supports ATI to let you start CICS tasks based on the presence of records in the queue, without user intervention or terminal input. The queue definition specifies:

- **TRAnsid**: what transaction should be started
- **Atifacility**: whether it should run at a terminal
- **Facilityid**: the name of that terminal
- **TRIggerlelvel**: the number of records required to trigger the task.

In this example, TRN1 writes to the TD queue MYTD defined to automatically initiate TRN2 at terminal PRNT when four records are present. When TRN2 starts, it can read the queue to pick up data passed there, and its SEND commands are routed to PRNT.

Atifacility=File is specified for initiation of non-terminal transactions. In this case, a userid has to be specified that will be associated with these transactions.

Indirect Queues

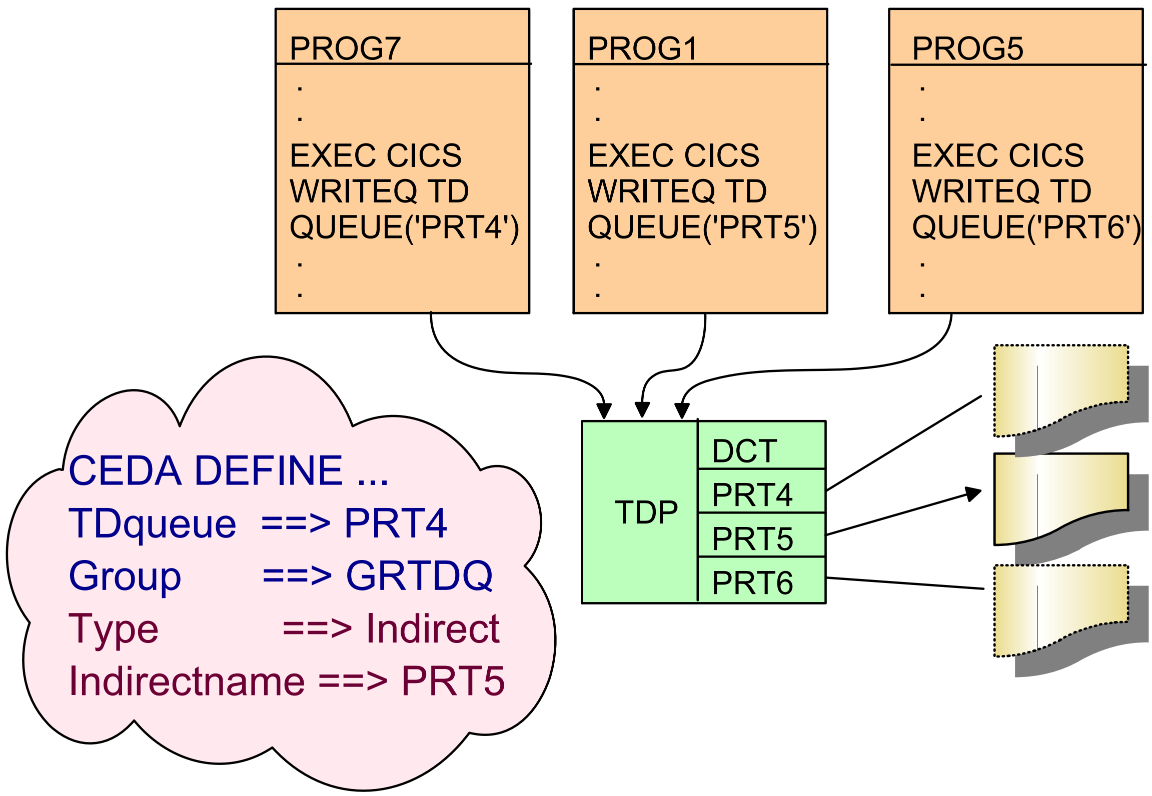


Figure 8-6. Indirect Queues CI207.0

Notes:

- An application was designed to use three printer terminals, but only one (PRT5) was available when the application was installed.
- To avoid either rewriting the programs or delaying the implementation:

  a. Define PRT5 as an INTRA queue.
  b. Define PRT4 and PRT6 as INDIRECT pointing to PRT5.

- When programs write to PRT4 or PRT6, the output is redirected to PRT5.
- When the physical printers are installed, modify the definitions to make PRT4 and PRT6 both INTRAs.

Types of Extrapartition Queues

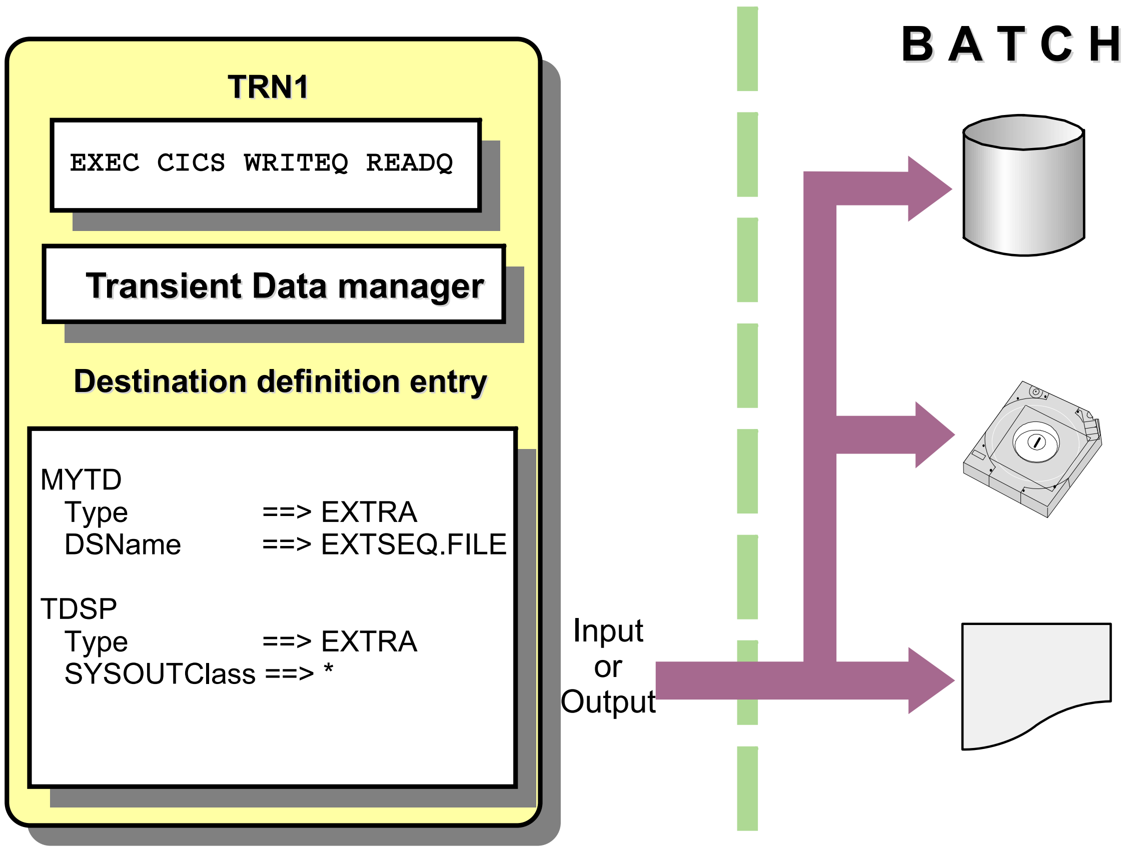


Figure 8-7. Types of Extrapartition Queues CI207.0

Notes:

- Each EXTRApartition queue represents a sequential data set, a member of a partitioned data set (since CICS TS 1.3), or an internal reader.
- You specify an EXTRA queue as either input or output, but not both.
- If an EXTRA queue is specified as output, it must be closed and reopened with a different queue name before it can be read. The open and close may be done with CEMT or an application program.
- DD statement in the CICS startup JCL is no longer required if the DSName or the SYSOUT class associated to the output is specified within the TD queue definition.
- Any values coded for the BLKSIZE and RECFORM in the RDO definitions override JCL specifications.
- EXTRA TD queues cannot be defined as recoverable, and do not support ATI.

CICS Uses of Transient Data - the DFHDCTG RDO Group

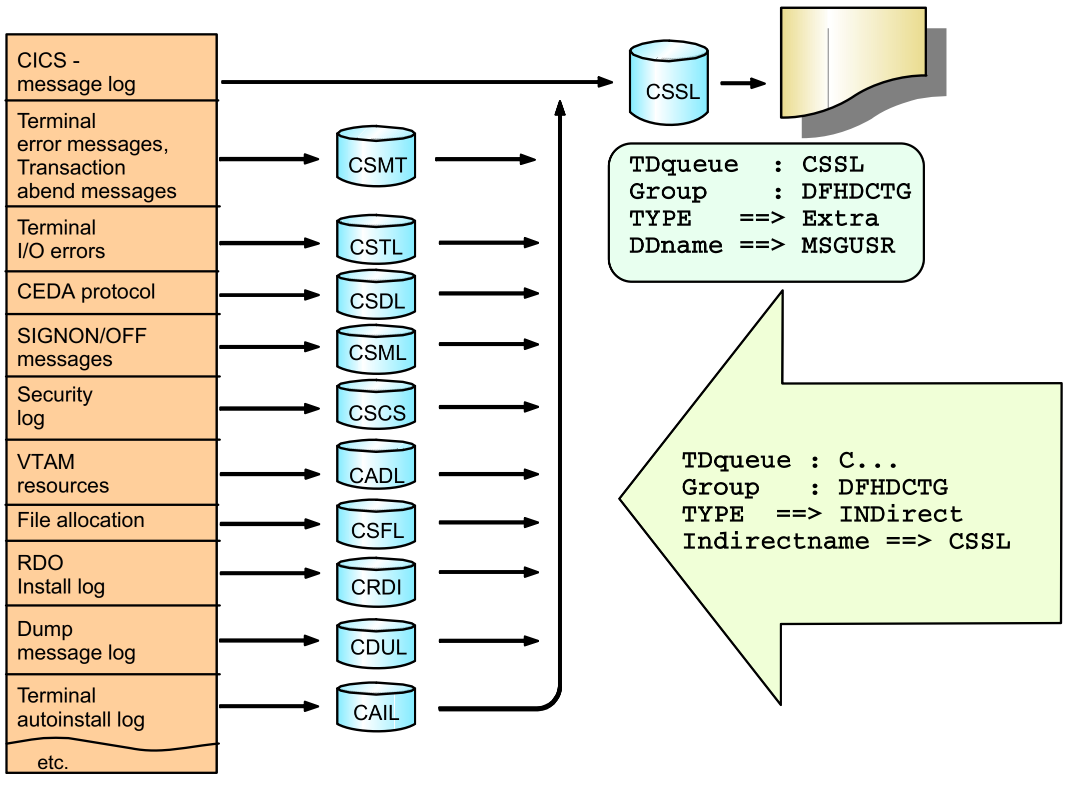


Figure 8-8. CICS Uses of Transient Data - the DFHDCTG RDO Group CI207.0

Notes:

- These sample (“default”) definitions are supplied in RDO group **DFHDCTG**, which is the first group referenced by the CICS-supplied list DFHLIST in order to have the transient data queues installed as soon as possible.
- Information placed here is often helpful in performing problem determination.
- Refer to the “Transient Data Queues” chapter of the *CICS Resource Definition Guide, SC34-6430*, for information defining the queues CICS uses, and the type of information written to each queue.
- Transient data queue definitions cannot be over-installed during startup.
  But, unlike all other CICS-supplied DFH* groups, **the DFHDCTG resource definition group is not protected**, so its entries may be changed, if required, and additional TDQ resources may be created within this group, though this is not recommended.
  Be sure to create a copy of the original definitions before applying changes to them!

8.2 Temporary Storage Queues

Types of Temporary Storage

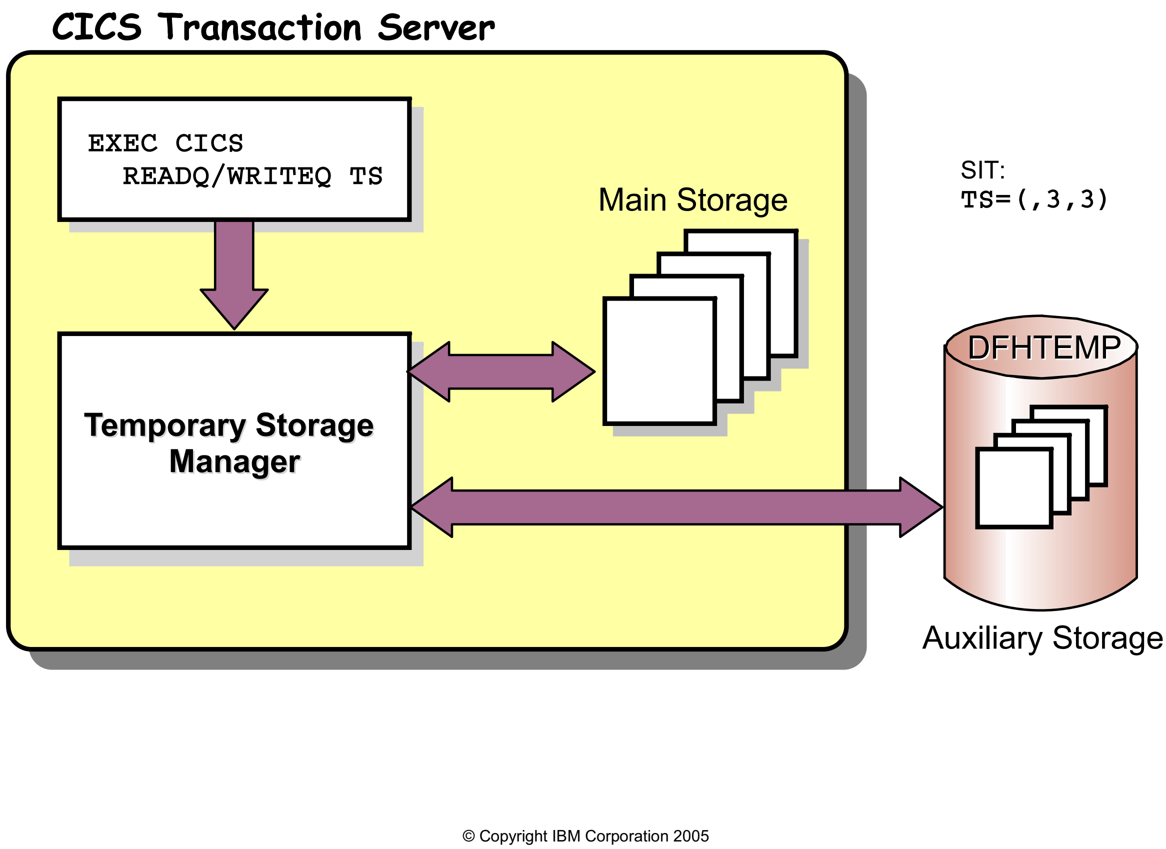


Figure 8-9. Types of Temporary Storage CI207.0

Notes:

- CICS TS builds the queue pointers dynamically the first time an **EXEC CICS WRITEQ TS** is issued using a unique queue name. The queue names do not have to be predefined.
- TS queue names are one to sixteen characters long.
- Programs can read TS queues sequentially or directly. Reading from TS is non-destructive, thus multiple read is possible.
- TS queue data may also be updated or deleted.
- Temporary storage queues may be defined or requested by the application to reside in either:

  MAIN Storage allocated in Extended DSA (ECDSA)

  AUX Storage allocated on a VSAM ESDS with DDname DFHTEMP

  Main storage TS queue records are lost when CICS shuts down, whereas auxiliary TS queues are perceived when CICS is recycled.

- Use the TS=(,b,s) parameter in the SIT to define the number of buffers and strings for AUX TS. The defaults are three of each, the maximum 32,767.
  If no (or “0”) buffers are specified, all requests are sent to main storage.
- The “Defining the Temporary Storage Data Set” chapter of the *CICS System Definition Guide, SC34-6428,* contains information for DFHTEMP allocation and JCL for use with CICS.
- Each TS queue name must be unique. It is recommended to establish standards to make the queue names unique. Queue names built with the name of the terminal may be a problem when migrating applications to an IP CICS Sockets environment, as no TERMIDs are associated to the tasks.

TST Macro Table and RDO TSMODEL Object

TST Macro Table	TSMODEL definition

`DFHTST TYPE=RECOVERY,`
`DATAID=REC`
	`DEFINE TSMODEL(TSREC)`
	`PRefix ==> REC`
	`RECovery ==> Yes`
	`Location ==> Aux`

`DFHTST TYPE=SECURITY,`
`DATAID=SEC`
	`DEFINE TSMODEL(TSSEC)`
	`PRefix ==> SEC`
	`Security ==> Yes`
	`Location ==> Main|Aux`

`DFHTST TYPE=REMOTE,`
`DATAID=REM1,`
`SYSIDNT=QOR1`
	`DEFINE TSMODEL(TSREM)`
	`PRefix ==> REM`
	`REMOTESystem ==> QOR1`

Figure 8-10. TST Macro Table and RDO TSMODEL Object CI207.0

Notes:

- Temporary storage queues that are created dynamically on behalf of an application program's access always have the default characteristics of being:
  - Local to the region where the application runs (unless the `SYSID` option is used by the application)
  - Nonrecoverable
  - Not access-protectable by security profiles
- If one of these characteristics is required as non-default, a definition has to be provided, either as a Temporary Storage Table (TST) entry or as an RDO TSMODEL resource definition type, which was introduced with CICS TS 1.3.
- Queue names specified within TST/TSMODEL definitions are always handled as generic names. So, the sample above reads as follows:
  - TS queues named REC* will be recoverable, and therefore will always go to auxiliary storage.

- TS queues named SEC* may be secured by the use of CICS resource security profiles.
- Requests for TS queues named REM1* will be function-shipped to region QOR1.

- The TST macro table is one of the very few that is still supported by CICS TS for z/OS. But it is recommended that you use the RDO definition method, as the TSMODEL definition provides attributes that are not supported by the macro:
  - The `Location` option, which determines whether a TS queue is created in main storage or in the DFHTEMP disk data set (auxiliary storage), is available only with the RDO definition.

    If a TS queue is created with no matching TSMODEL definition active, the “Auxiliary” location takes effect as the default, unless the application uses the `MAIN` option when referencing the queue for the first time.

    If a TSMODEL definition is there, the location option always is taken from the TSMODEL, no matter what may be specified by the application.
  - The prefix specified in a TSMODEL resource definition may be up to 16 characters in length, while the DATAID value provided by the macro definition is limited to 8 characters.

    Note that the name of the TSMODEL definition itself can only be 8 characters in length, and it need not be correlated to the prefix specified in any way.

- Refer to the *CICS Resource Definition Guide, SC34-6430*, for details.

- Be careful when dynamically `INSTALL`ing and `DISCARD`ing TSMODELs!
  This might do harm to application processing, because changes to TSMODELs can be activated at any time, but they will not affect existing TS queues that have been created based on the “old” definition.

Application Uses

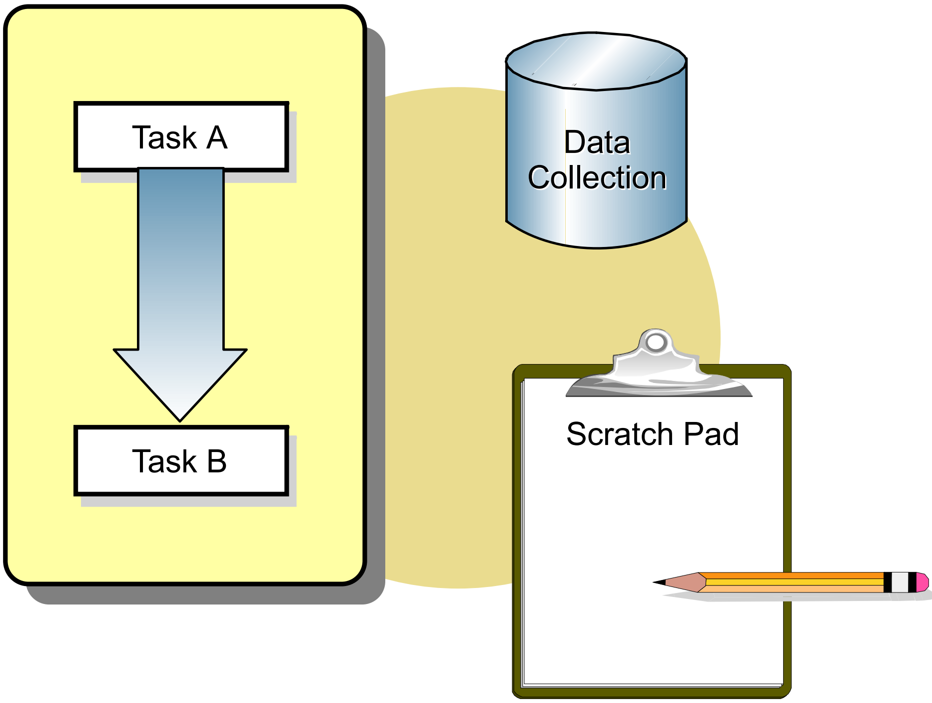


Figure 8-11. Application Uses CI207.0

Notes:

Common application uses for temporary storage include:

- Passing data between pseudoconversational tasks
- Collecting data, especially data pertinent to a terminal
- A scratch pad, for example, teller totals for debits and credits.

CICS Uses of Temporary Storage

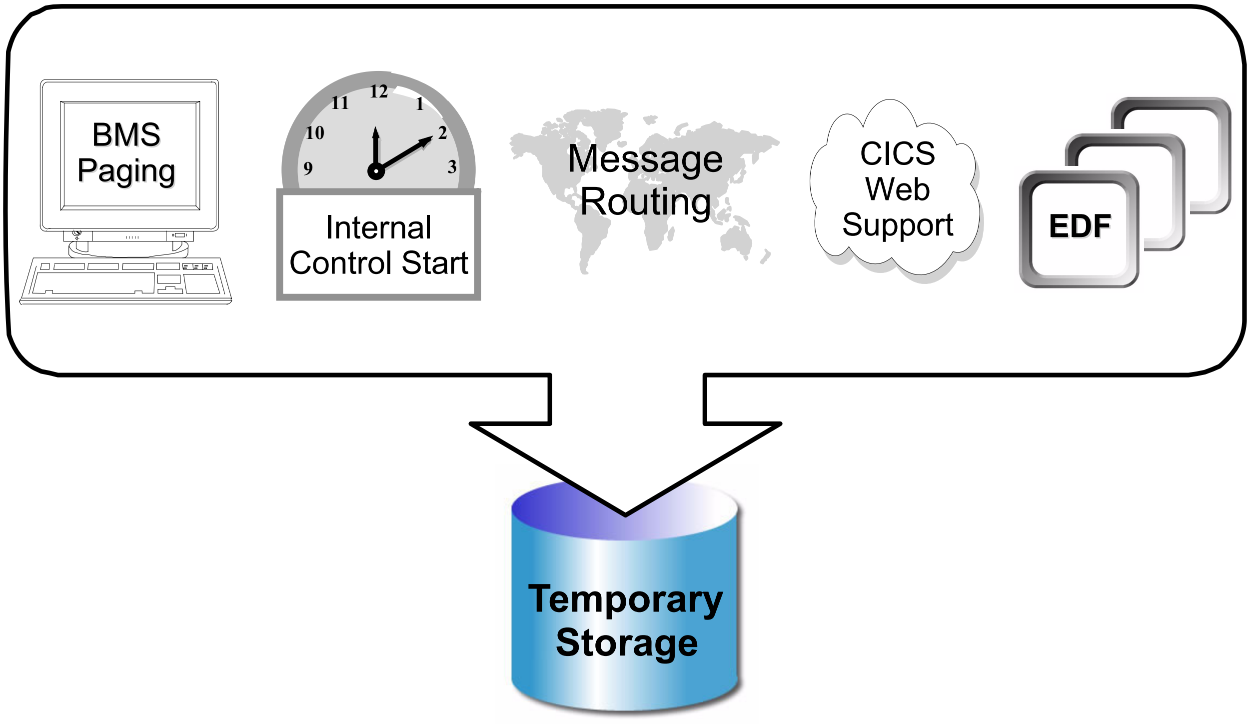


Figure 8-12. CICS Uses of Temporary Storage CI207.0

Notes:

- Basic Mapping Support (BMS) uses TS for saving output pages and routed messages. These TS queues use names starting with non-printable characters.
- The Interval Control Program (ICP) uses TS to pass data to STARTed transactions. The names of these TS queues start with the characters `DF`, which for this reason must not be used by applications.
- When using the CEDF transaction, each EXEC CICS command will cause debugging information to be written to a TS queue.
- If VS COBOL II is used and the CICS Translator is run with the DEBUG option, CICS writes a brief dump in the event of a transaction abend.
  The queue name is `CEBRtttt`, where `tttt` is the terminal ID.
- The CICS Web Support Function by default stores data received from the Web in a TS queue before the components are started that process the request. These queues are based on the DFHWEB TSMODEL definition provided in group DFHWEB.

Inquiring TSQs

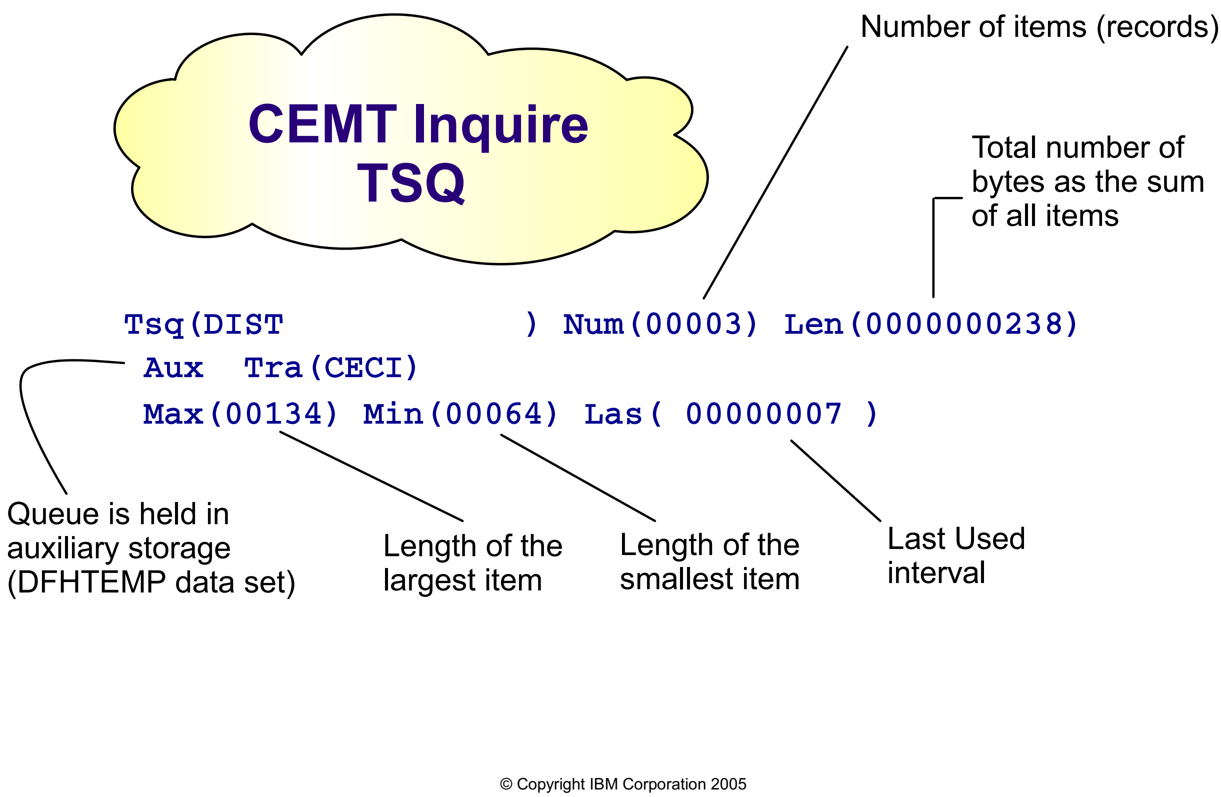


Figure 8-13. Inquiring TSQs CI207.0

Notes:

- The CEMT INQ TSQ command returns the names of the TS queues currently owned by a region, together with some valuable information, as shown on the visual.
- Of particular interest are:
  - The name of the transaction that caused the creation of the TS queue: this may (should) indicate to which application the queue belongs.
  - The Last Reference timer/counter: it identifies the time (in seconds) since the queue was last referenced, either by a READQ or WRITEQ operation.

CICS-Supplied Transaction CEBR

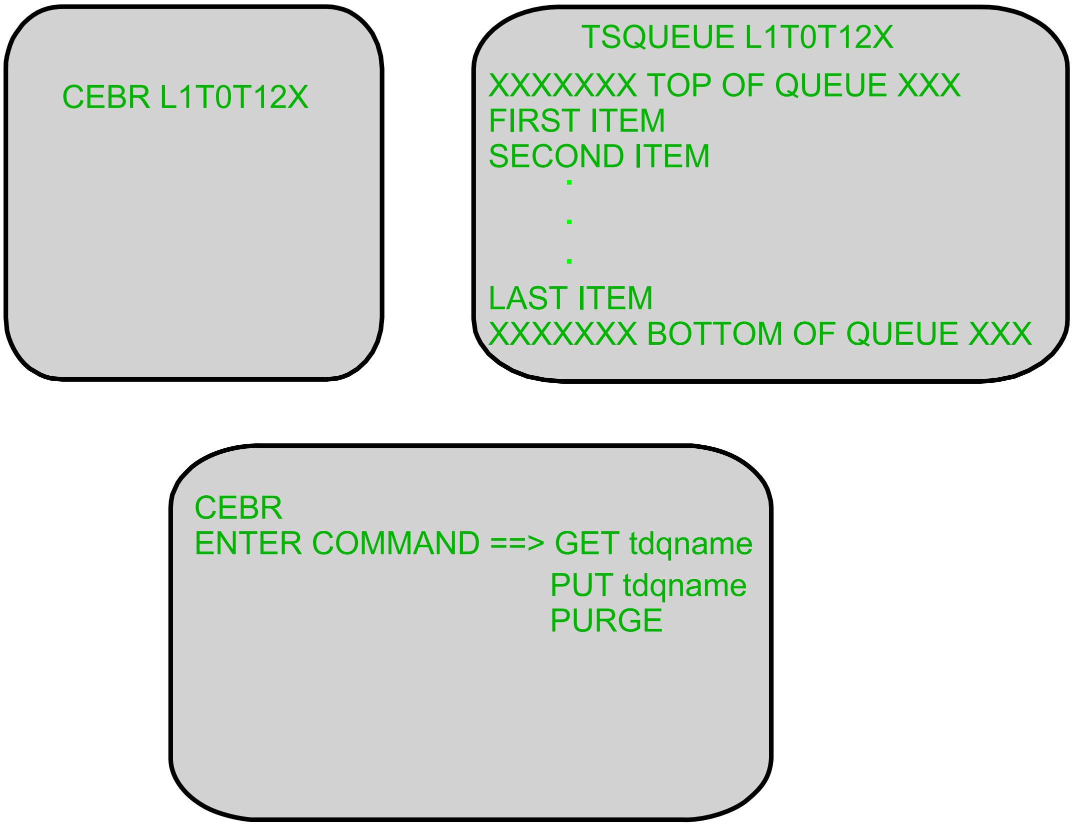


Figure 8-14. CICS-Supplied Transaction CEBR CI207.0

Notes:

- CICS-supplied transaction CEBR allows browsing of named Temporary Storage Queues displaying one item per screen line. Shifting and scrolling are supported.
- Subfunctions GET and PUT transfer the contents of a named **Transient Data** Queue to the TS Queue and vice versa (caution: TDQUEUE is emptied by reading!).
- PURGE specifies the deletion of the Temporary Storage Queue.
- Handle with care, if you do anything other than pure browsing!

Transient Data and Temporary Storage Comparison

Facility	Temporary Storage	Transient Data
Queue definition	Dynamic, by application or static TST, TSMODEL	Static, via RDO
Record Retrieval	Direct and Sequential	Sequential only
Multiple Reads	Yes	No
Use of Main Storage	Yes, by option "MAIN"	No
Use of DASD	Yes, by option "AUX"	Yes (always)
Non-CICS access	No	Yes (Extra)
Redirect Output	No	Yes (Indirect)
Recovery	Yes (AUX only)	Yes (Intra only)
Automatic Task Initiation	No	Yes (Intra)

Figure 8-15. Transient Data and Temporary Storage Comparison CI207.0

Notes:

The main differences between Transient Data and Temporary Storage are:

- TD destinations must be predefined, with RDO.
- Since TS does not delete records as they are retrieved, applications must handle purging.
- ATI with intrapartition transient data is triggered by the presence of records in the queue. TS does not actually cause ATI, but is used by EXEC CICS START to store passed data.

Queuing (Summary)

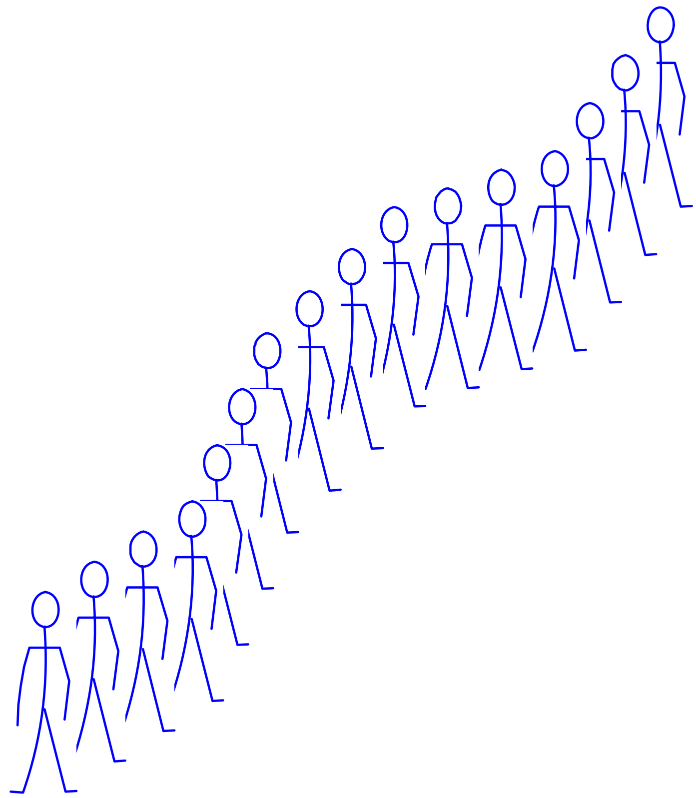


Figure 8-16. Queuing (Summary) CI207.0

Notes:

- CICS offers two queuing facilities: Transient Data and Temporary Storage.
- Transient Data INTRApartition queues:
  - Can only be accessed by CICS programs
  - Reside on a VSAM ESDS named DFHINTRA
  - Can cause automatic transaction initiation when a specified trigger level is reached.
- Transient Data supports access to QSAM files as EXTRApartition queues.
- MAIN TS queues are built in Extended DSA, while AUX queues are built on a VSAM ESDS data set named DFHTEMP.
- SIT parameters specify the number of buffers and strings for DFHINTRA and DFHTEMP.
- CICS makes extensive use of the queuing facilities.
- CEBR is a CICS-supplied transaction to browse and purge TS queues.

Unit Summary

Having completed this unit, you should be able to:

- Describe the CICS Transient Data and Temporary Storage queuing facilities
- Contrast intrapartition and extrapartition queues
- Describe Automatic Transaction Initiation (ATI)
- Define transient data queues using RDO to support applications using intrapartition, extrapartition, and indirect queues, and ATI
- Contrast main and auxiliary Temporary Storage
- List CICS uses of queuing facilities
- Name the CICS system data sets used for queues
- Specify the required SIT parameters to support queuing

Figure 8-17. Unit Summary CI207.0

Notes:

Unit 9. CICS Intercommunication Services

What This Unit Is About

This unit covers the functions that support the connection of devices that use the SNA LU6.2 protocol to access CICS, and what has to be defined to support them. It also provides an overview of CICS client support, regarding network-related issues.

More and more, CICS is connected to workstations of whatever kind in order to communicate with the non-CICS systems and applications running on those workstations.

The basic approach to do this was, and still is, to use the SNA LU6.2 protocol that offers all the options necessary to create powerful distributed applications with access to CICS Transaction Server.

What You Should Be Able to Do

After completing this unit, you should be able to:

- Provide definitions to support the connection of LU6.2 devices
- Configure the LU6.2 support of the CICS terminal autoinstall function
- Provide definitions to support connection of TCP/IP partner systems using the HTTP protocol, the IIOP protocol, or the ECI protocol
- Provide definitions to support MRO connections between CICS systems
- Provide definitions to support EXCI connections from batch programs
- Describe the dynamic routing facilities provided by CICS
- Provide definitions required to access remote CICS resources
- Provide definitions to support connection of Web Services requesters using the SOAP protocol
- Provide definitions to support connection to Web Services providers using the SOAP protocol

How You Will Check Your Progress

- Instructor questions

References

CICS Transaction Server for z/OS Version 3.1 Installation Guide, GC34-6426

CICS System Definition Guide, SC34-6428

CICS Resource Definition Guide, SC34-6430

Unit Objectives

After completing this unit, you should be able to:

- Provide definitions to support the connection of LU6.2 devices
- Configure the LU6.2 support of the CICS terminal autoinstall function
- Provide definitions to support connection of TCP/IP partner systems using the HTTP protocol, the IIOP protocol or the ECI protocol
- Provide definitions to support MRO connections between CICS systems
- Provide definitions to support EXCI connections from batch programs
- Describe the dynamic routing facilities provided by CICS
- Provide definitions required to access remote CICS resources
- Provide definitions to support connection of Web Services requesters using the SOAP protocol
- Provide definitions to support connection to Web Services providers using the SOAP protocol

Figure 9-1. Unit Objectives CI207.0

Notes:

CICS Communication Services

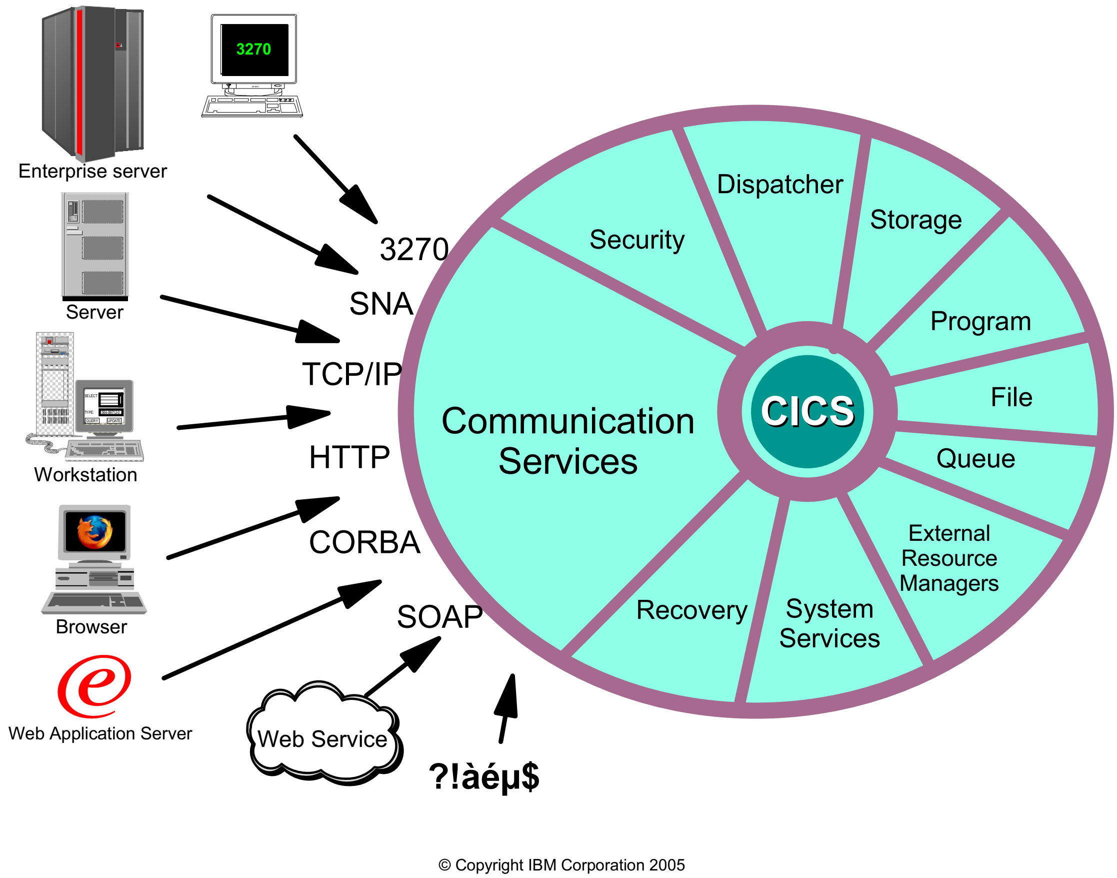


Figure 9-2. CICS Communication Services CI207.0

Notes:

CICS provides extensive partner systems communication services facilities using a wide range of communication protocols, from old SNA-based protocol to the most recent SOAP protocol.

SNA partners, in other words enterprise or distributed server applications, use the LU6.2 SNA protocol to communicate with CICS applications. CICS can play the role of a server or a client.

Partners using the TCP/IP transport may use different protocols to communicate with CICS applications. The protocol which is used determines the quality of service. It ranges from poor QoS TCP/IP sockets protocol to high QoS CORBA J2EE or Web Services SOAP protocols.

You may wonder about the **?!àéµ$** protocol! This is just to make the point that CICS will support any future emerging market chosen protocol. It has been doing this for more than 30 years now.

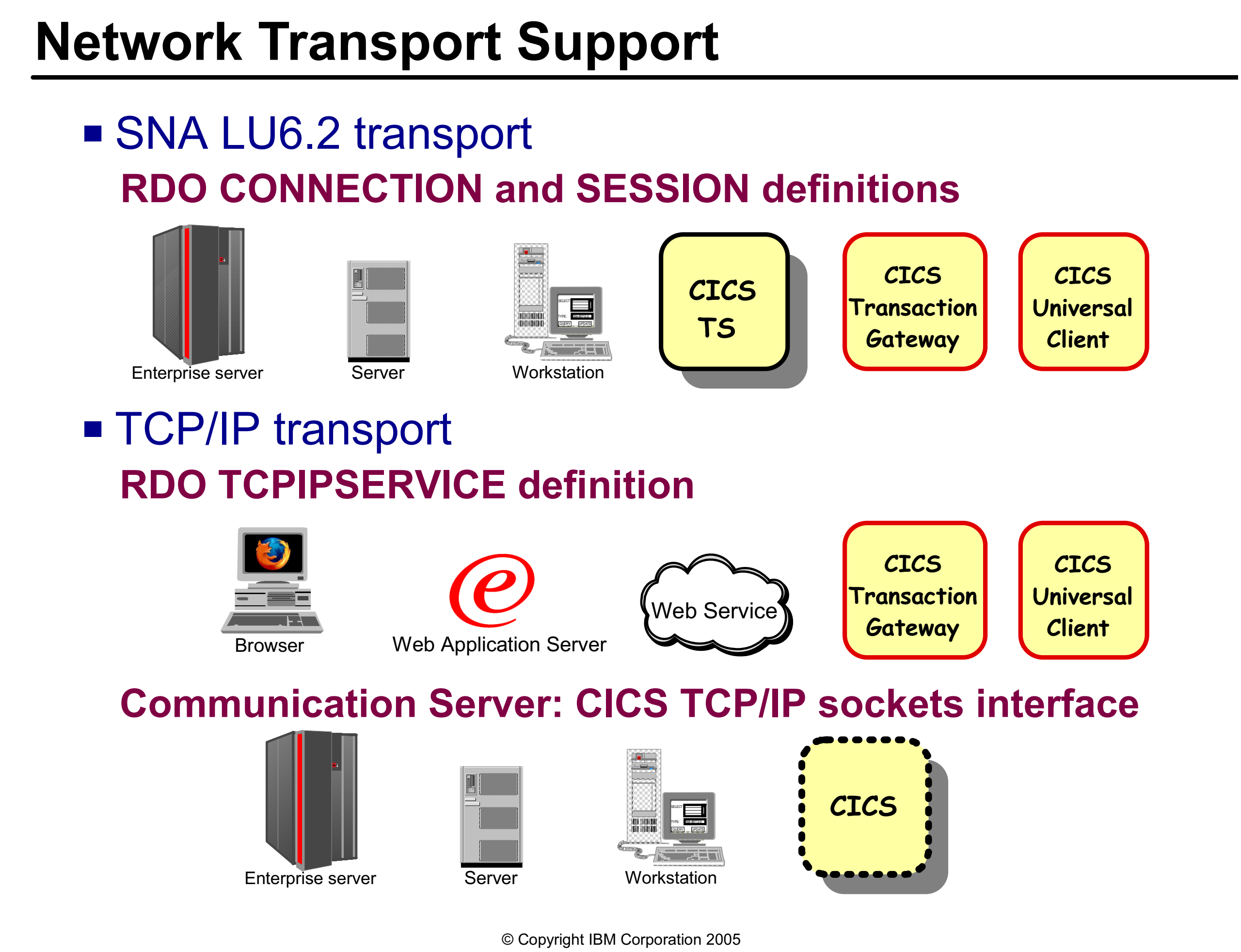


Figure 9-3. Network Transport Support CI207.0

Notes:

CICS supports SNA- or TCP/IP-based protocols.

SNA supplies high quality of service program-to-program communication based on the LUType 6.2 (LU6.2) protocol which provides Advanced Program-to-Program Communication (APPC) support. Partner applications can be:

- Enterprise Server applications (MVS/APPC or IMS/TM)
- Distributed UNIX or Intel Server applications (CPI-C programs)
- CICS applications using the CICS Distributed Transaction Processing APIs
- CICS connectors applications using the CTG or CUC APIs

LU6.2 links are defined to CICS using the RDO CONNECTION and SESSION resource definitions.

TCP/IP does not supply high quality of service program-to-program communication. TCP/IP supplies a sockets interface whose support is supplied by Communication Server for z/OS as an External Resource Manager interface. Any TCP/IP socket application, including CICS applications, can communicate with CICS.

TCP/IP quality of service comes with standard protocols such as Internet HTTP or CORBA IIOP. CICS supports both of them along with its own private External Call Interface (ECI) used by the CTG and CUC connectors.

- HTTP protocol can be used from Web browsers, HTTP client applications, or Web Services applications.
- CORBA IIOP protocol can be used from J2EE Enterprise Java Bean (EJB) applications running in a Web Application Server such as WebSphere.

TCP/IP links are defined using the RDO TCPIPSERVICE resource definition.

When SNA LU6.2 CONNECTION/SESSION definitions are used for both inbound and outbound communication, TCP/IP TCPIPSERVICE definitions are used for inbound communication only.

9.1 LU6.2 Support

LU6.2 Definitions

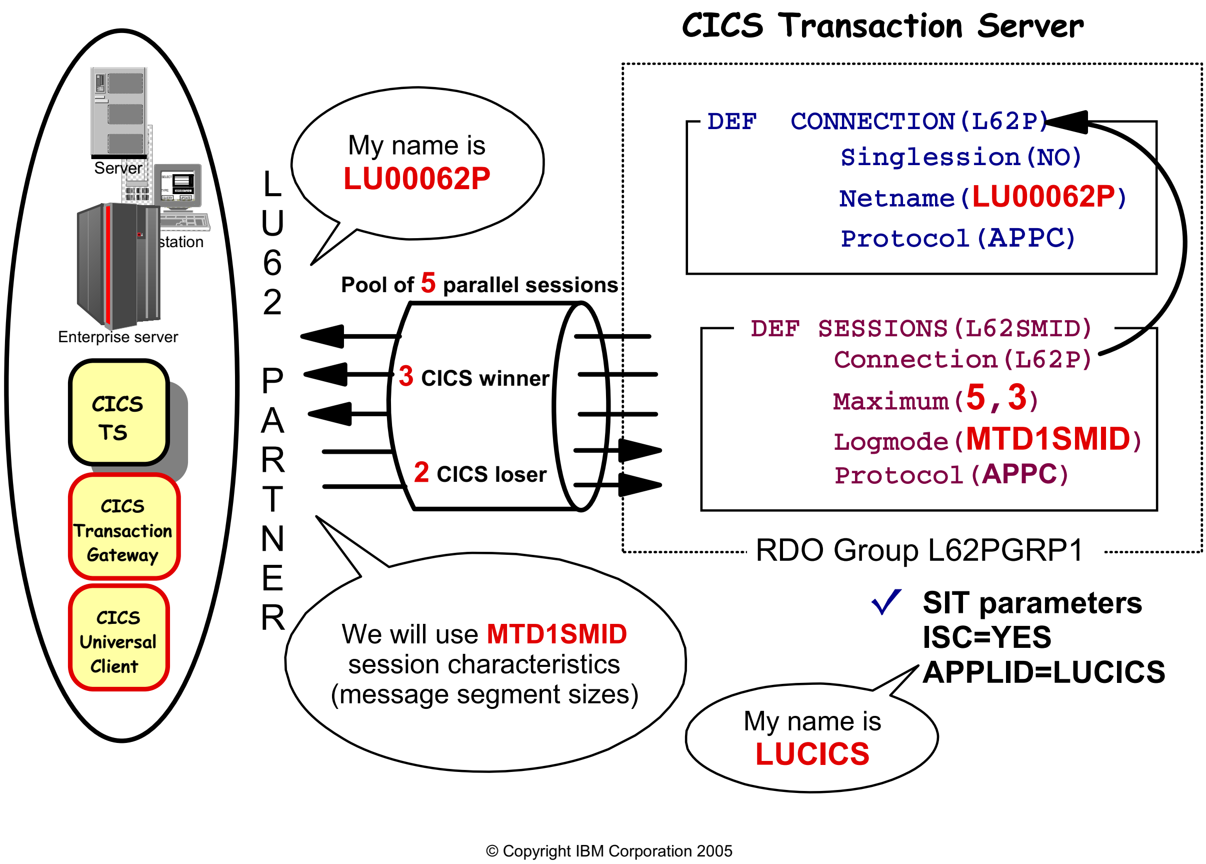


Figure 9-4. LU6.2 Definitions CI207.0

Notes

Parallel Session connections are defined using the **CONNECTION** and **SESSIONS** RDO resource types.

The CONNECTION definition defines the partner system to CICS. Also, it supplies its SNA network name and the LU6.2 security features to be used.

The SESSION definition defines the LU6.2 characteristics of the conversations (or sessions) exchanged between the two partners. Also, it supplies the maximum number of parallel conversations maintained between both systems along with the number of CICS outbound sessions. The LOGMODE parameter defines the network session characteristics such as the message segments length.

Multiple SESSION groups may be used by a single CONNECTION. This allows the definition of different conversation characteristics for different applications using the same LU6.2 link.

In order to use LU6.2 links, the SIT ISC=YES parameter must be specified. The CICS SNA network name is defined by the SIT APPLID parameter.

Controlling LU6.2 Connections

- CEMT I CON(L62P)

  Con(L62P) Net(LU00062P) Ins Acq Vta Appc

- CEMT I TER(*) NET(LU00062P)

  Ter(-AAA) Pri(000) Aut Ins Ati Tti
  Net(LU00062P) Acq Rem(L62P)

  Ter(-AAB) Pri(000) Aut Ins Ati Tti
  Net(LU00062P) Acq Rem(L62P)

  + One Ter(-...) entry per SESSION

- CEMT I MODE(*) CON(L62P)

  Mod(SNASVCMG) Con(L62P) Max(002) Ava(002) Act(001)

  Mod(LU00062P) Con(L62P) Max(005) Ava(005) Act(003)

Figure 9-5. Controlling LU6.2 Connections CI207.0

Notes:

- At (RDO) installation time, for each session of a LU6.2 connection, a TCT System Entry (TCTSE) is created with a *termid* allocated that is determined dynamically, both for single and parallel session connections.
  - The *termids* allocated to the LU6.2 sessions are named -AAA, -AAB, ... and are associated with the VTAM netname that is defined in the CONNECTION or the TERMINAL definition.
  - When a transaction is run, the session used becomes the task's principal facility.
- A set of sessions defined by a SESSIONS object, with the operational characteristics specified by its LOGMODE parameter, is called a **modeset**.
  - For an active LU6.2 connection, its **modeset(s)** can be inquired and acted on with the name of the LOGMODE specified in the SESSIONS or TERMINAL definitions.
  - In addition to the logmode(s) defined, for parallel session connections, LU6.2 system services modeset **SNASVCMG** will always be active, which is the VTAM "administration session".

LU6.2 Autoinstall

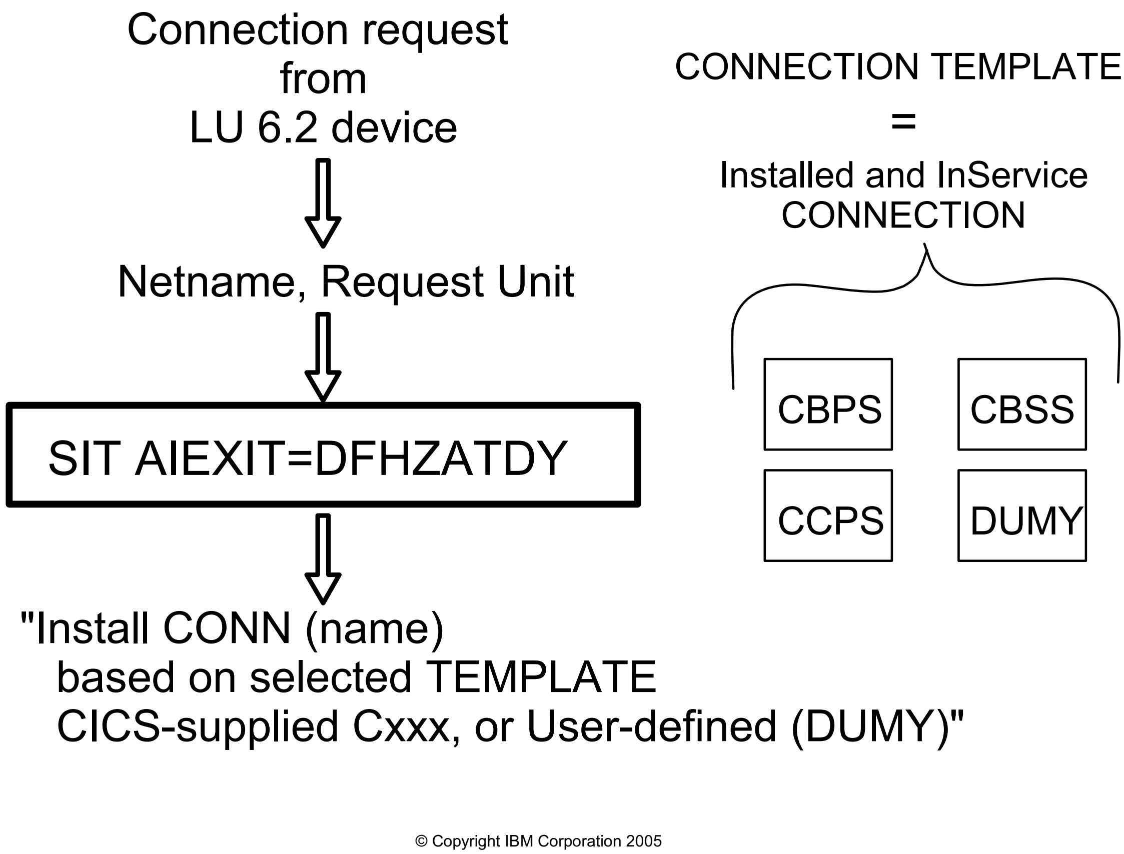


Figure 9-6. LU6.2 Autoinstall CI207.0

Notes:

Terminal autoinstall is supported for LU6.2 devices, with some additional requirements for the autoinstall exit specified with the SIT parameter `AIEXIT`.

- The CONNECTIONs dynamically created are based on so-called **templates** instead of autoinstall models. Any CONNECTION installed may be used as a template; there is no special identification of “models” as for simple terminals. You are advised not to use ‘real’ connection definitions, but to define dummy ones. CICS supplies sample template definitions in the RDO group DFHAI62.
- There is no preselection of a fitting template by the CICS system code; the exit has to determine everything based on VTAM netname and BIND information.
- Besides determining the connection ID, the AI exit is allowed to change the parameters of the CONNECTION-template to construct the new dynamic CONNECTION definition.
- CICS supplies a sample AI exit names DFHZATDY whose RDO definition is supplied in the sample DFHAI62 group.

9.2 TCP/IP Support

TCP/IP Transport Definitions

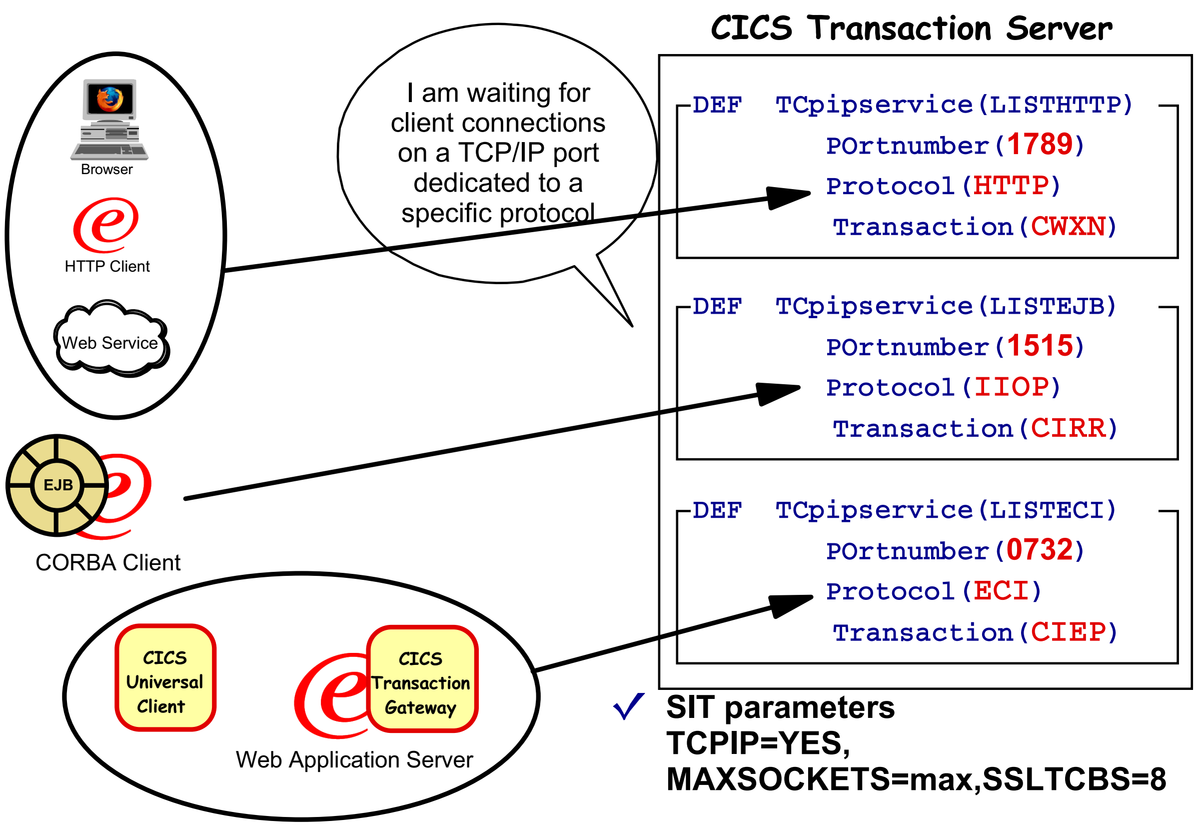


Figure 9-7. TCP/IP Transport Definitions CI207.0

Notes:

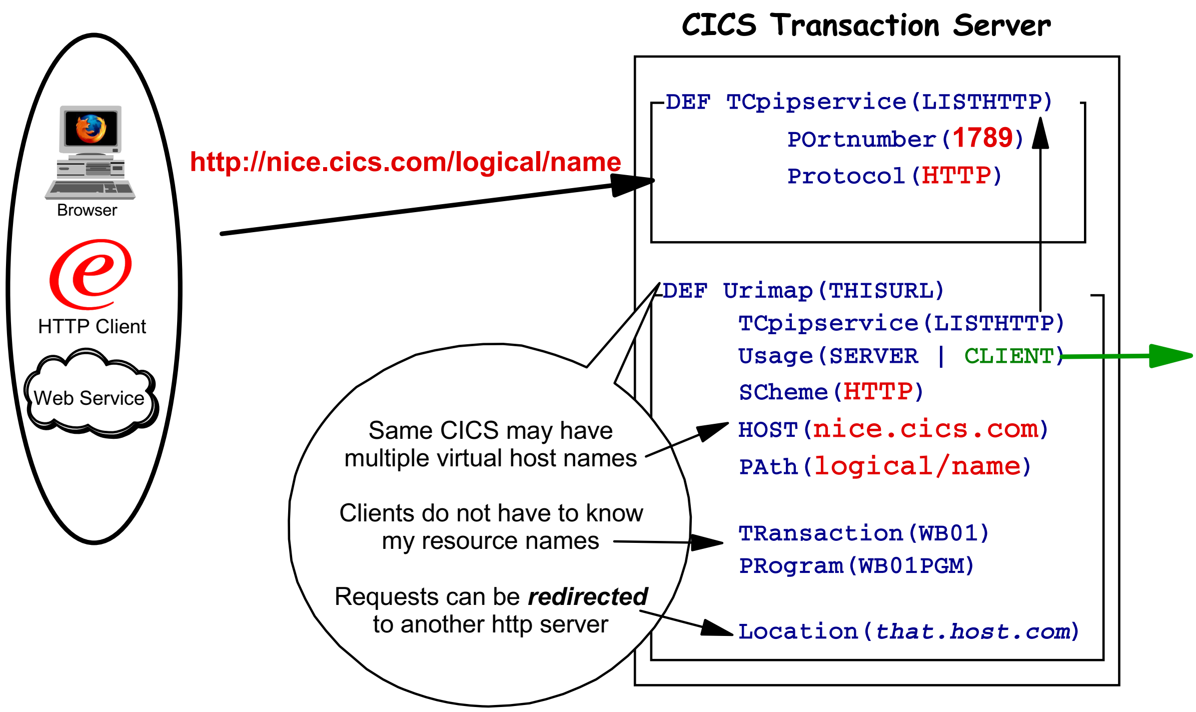
In order to support TCP/IP inbound connections, a TCP/IP server must be LISTENing on a PORT number. CICS listeners are defined by TCPIPSERVICE resource definitions which are dedicated to a specific protocol. CICS supports three different protocols:

1. **HTTP**, which is used by Web browsers, HTTP client applications, or Web Services requester applications, to connect to CICS applications.
2. **IIOP**, which is used by Java client applications, including J2EE Enterprise Java Beans, to connect to CICS applications.
3. **ECI**, which is a private CICS protocol used by the CTG and CUC connectors. ECI allows a client application to call a CICS program from a non-Java program (in other words, .NET application) or from a Java program.

TCP/IP support requires TCPIP=YES to be set in the SIT. MAXSOCKETS limits the maximum number of TCP/IP client connections which a single CICS region may support.

If the Secured Socket Layer (SSL) protocol is used, SSLTCBS specifies the maximum number of TCB dedicated to SSL certificate and encryption processing.

HTTP Transport Definitions



URIMAP can also be used to alias URLs on CICS HTTP CLIENT application requests

Figure 9-8. HTTP Transport Definitions CI207.0

Notes:

HTTP protocol makes use of Uniform Resource Locators (URLs) to identify the resources accessed. CICS supplies the URIMAP resource definition in order to map a CICS transaction code and a CICS program to a logical URL.

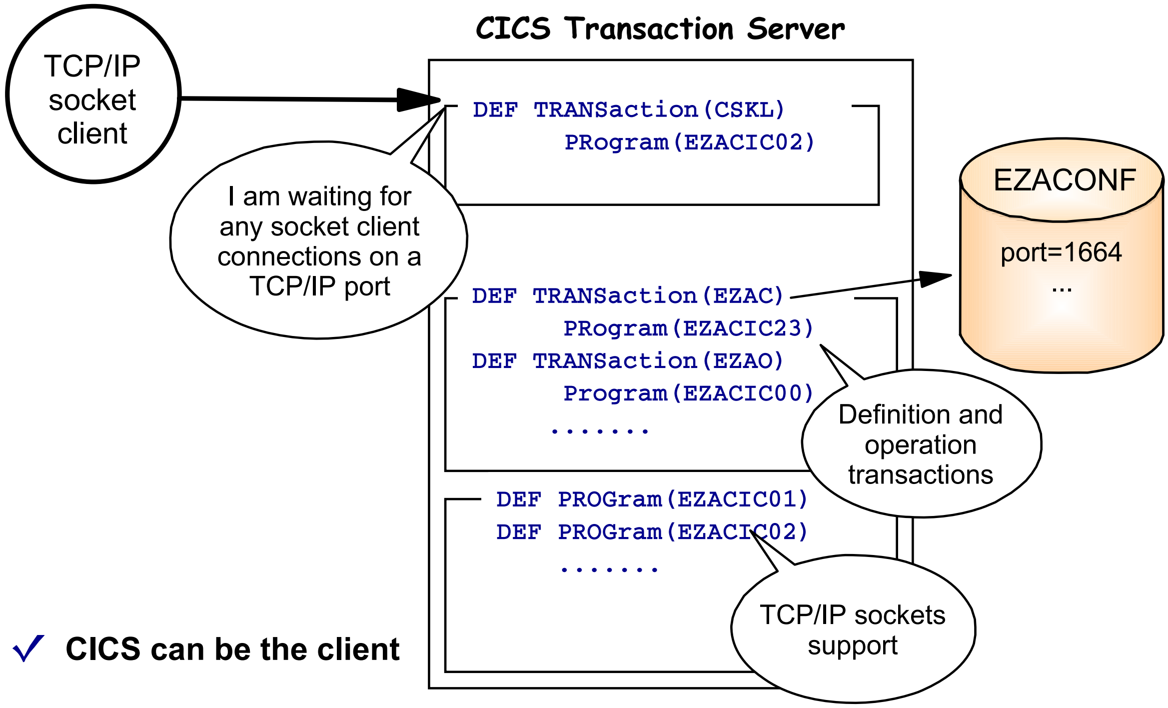
A URL is composed of a SHEME, which defines the protocol, a *host* name, which identifies the target TCP/IP server, and a *path*, which identifies the target resource in the form of a UNIX file system resource.

The same CICS may by accessed using different so-called *virtual host names*. In such a case, multiple URIMAP definitions will point to the same CICS resources.

The URIMAP definition may redirect the request to another HTTP server. In such a case CICS does not run any transaction.

CICS HTTP client applications may use the URIMAP resource definition from the CICS supplied HTTP APIs, in order to alias target HTTP server applications.

TCP/IP Sockets Definitions



- ✓ **CICS can be the client**
- ✓ **Pure Communication Server implementation, CICS is not aware of the socket activity**

Figure 9-9. TCP/IP Sockets Definitions CI207.0

Notes:

Communication Server for z/OS supplies the CICS Sockets Interface function.

CICS is not aware of the socket activity implemented by this function, and recognizes this function as an external resource manager such as DB2 or WebSphere MQSeries.

The CICS Sockets Interface function is a set of transactions and programs. The most important ones are:

- **CSKL**, which is the TCP/IP socket listener. It handles inbound client socket connection requests.
- **EZAC**, which is the configuration management transaction. The configuration is saved to a VSAM file named EZACONF.
- **EZAO**, which is the operating transaction. In other words, it allows the socket interface support to be stopped or started.

9.3 Multi-Region Operation (MRO)

Multiregion Operation (MRO)

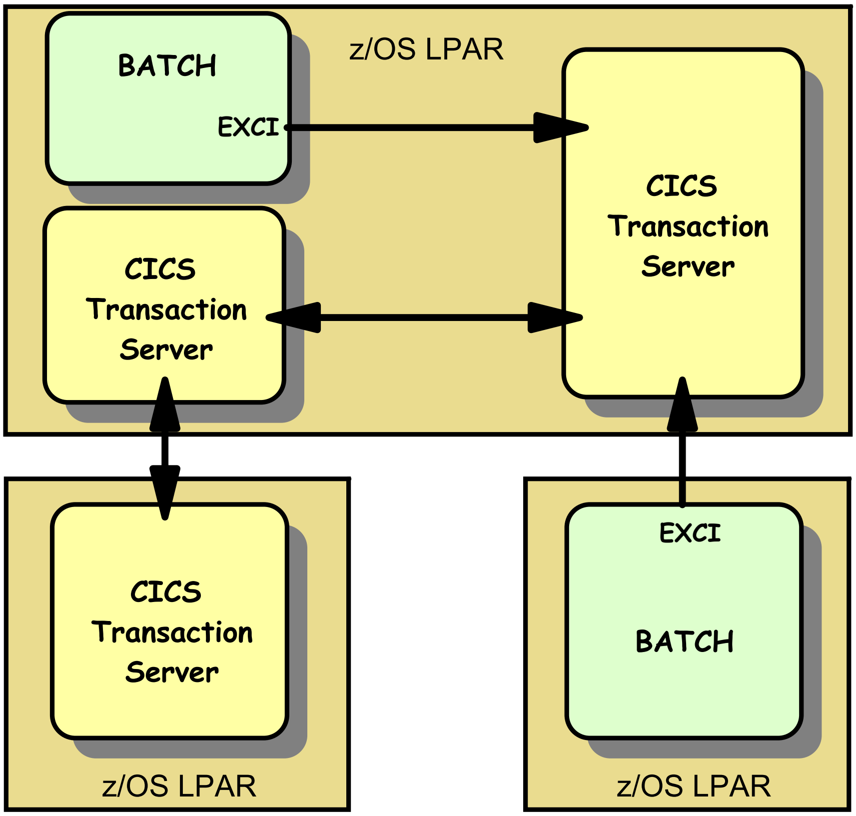


Figure 9-10. Multiregion Operation (MRO) CI207.0

Notes:

CICS Multi-Region Operation function (MRO) enables CICS systems that are running in the same z/OS image to communicate with each other using z/OS cross-memory services.

The CICS EXternal Call Interface (EXCI) function allows batch applications to link to CICS programs though an MRO link (named a EXCI PIPE). EXCI support is limited to batch outbound requests to CICS.

The z/OS cross-system coupling facility (XCF) provides a high-performance link between processes on z/OS images linked via ESCON channels, fiber optic links, or Channel-to-Channel (CTC) links.

MRO automatically uses XCF when the MRO partner is running in another z/OS image of a z/OS sysplex.

MRO Definitions

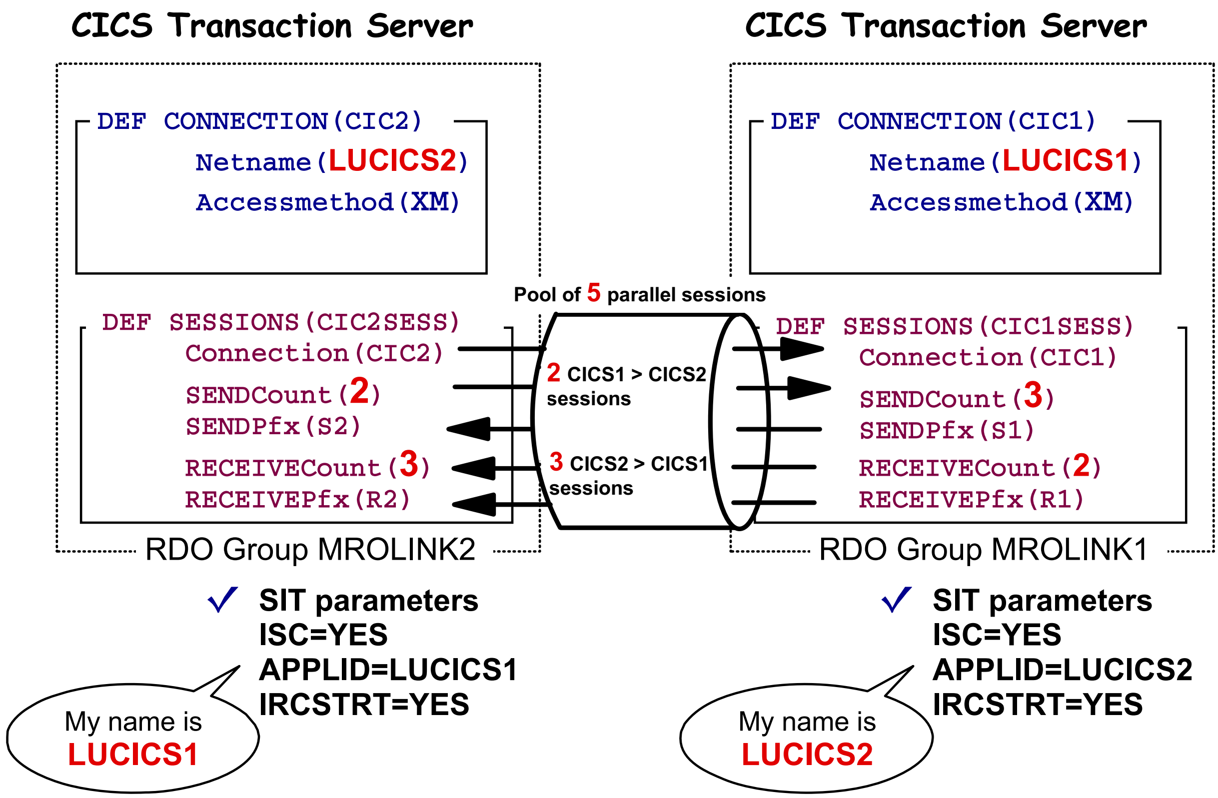


Figure 9-11. MRO Definitions CI207.0

Notes:

MRO definitions look very much like LU6.2 link definitions.

The CONNECTION resource definition defines the partner system and supplies its SNA network name (yes, indeed SNA) along with link security information, and the access method information (CrossMemory XM here).

The SESSION resource definition defines the conversation details such as the number of SEND and RECEIVE sessions shared by both partners. Notice that MRO session definition must supply a one- or two-character prefix to the session names.

Before two MRO partners may communicate, the Inter-Region Communication function must be opened. This is done by setting IRCSTRT=YES in the SIT.

Some code is shared with the LU6.2 support function, so don't be surprised to see that ISC=YES must be set in the SIT.

While LU6.2 supports multiple session definitions for a single connection definition, MRO supports only one session definition.

EXCI Definitions

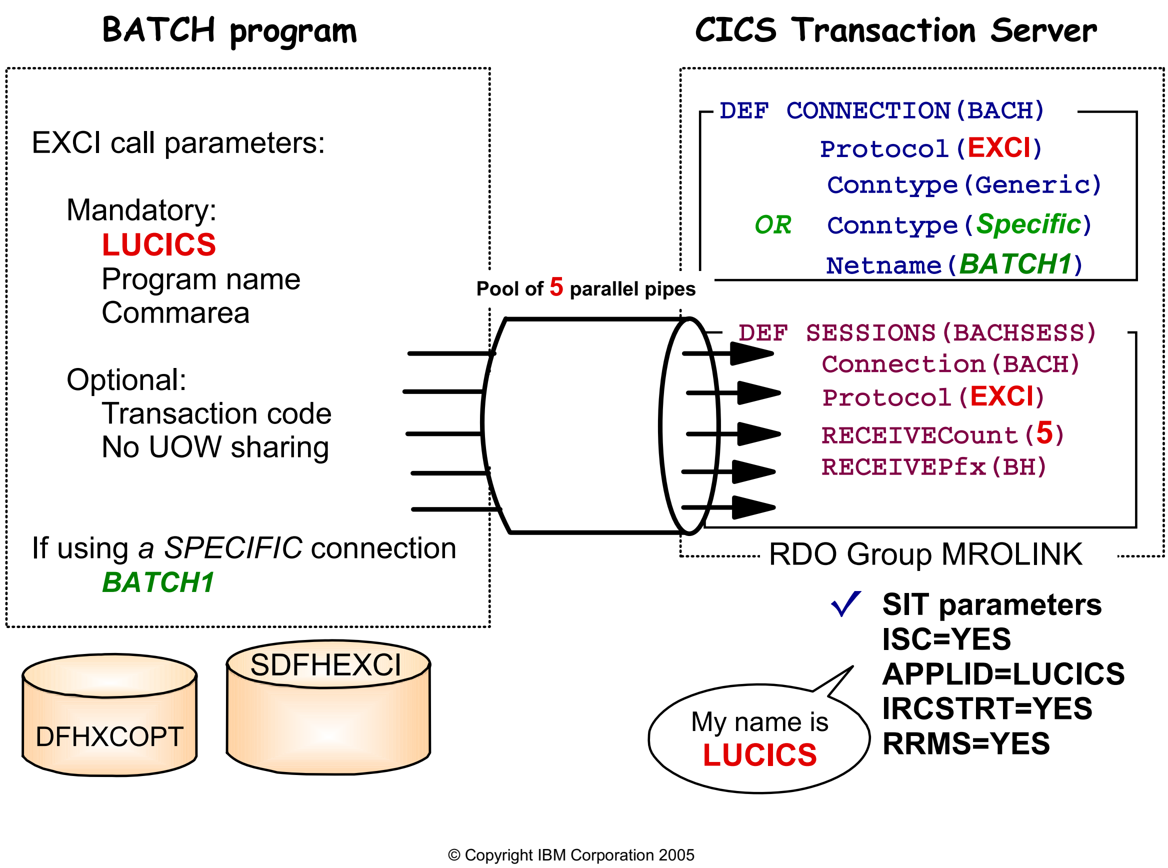


Figure 9-12. EXCI Definitions CI207.0

Notes:

EXCI is an MRO extension. The link between a batch program and CICS is defined by a CONNECTION/SESSION resource pair.

The CONNECTION defines the EXCI protocol, and the connection type to be used:

1. Generic means that a single CICS connection can be used by many batch partners.
2. Specific means that the CICS connection is reserved for a single batch partner which is identified by the NETNAME connection parameter.

The RRMS=YES SIT parameter indicates that CICS will participate in any unit of recovery shared with the batch program as required. In such a case, the z/OS Resource Recovery Manager (RRM) will be used to synchronize both partners.

There is a SEND session defined, as EXCI is a one-way communication link.

The batch application uses the EXCI support API supplied by the SDFHEXCI library. Some options can be defined in the DFHXCOPT assembler table; these options include trace and dump options.

9.4 CICS to CICS Intercommunication Services

Multiple CICS Regions Benefits

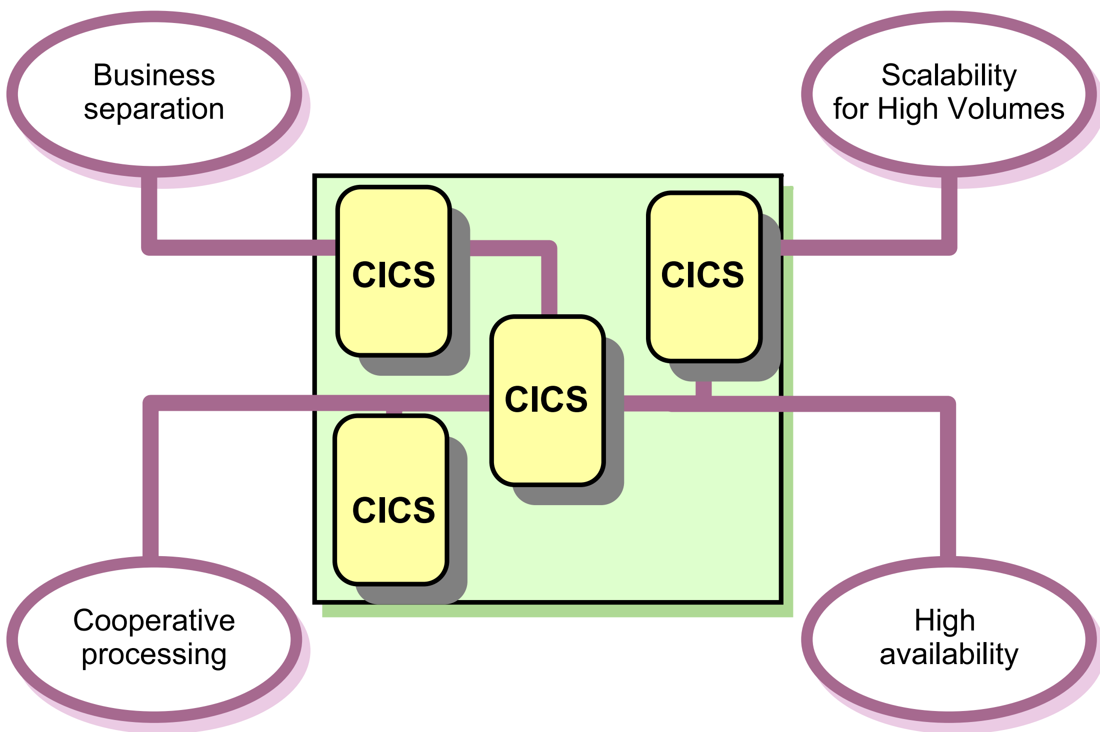


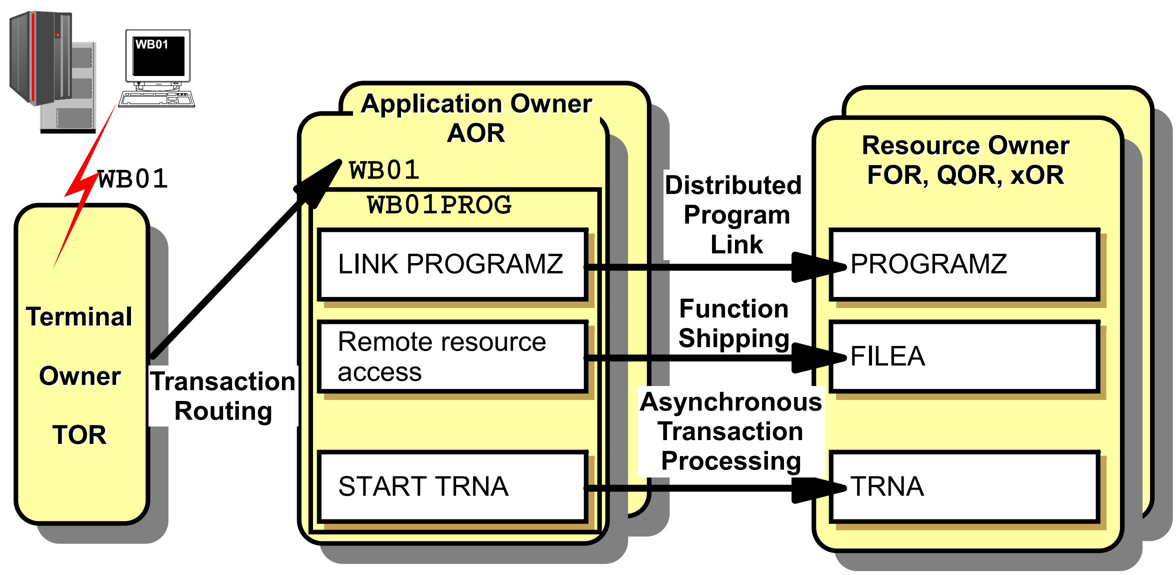
Figure 9-13. Multiple CICS Regions Benefits CI207.0

Notes:

SNA and TCP/IP are not the only ways to connect two CICS systems. CICS supplies the Multi-Region Operation function in order to create CICS complexes or CICSPlexes, and to address issues such as high availability or scalability.

To users or external client applications, the combined set of CICS regions may appear like a single server region.

CICS to CICS Intercommunication Services



A terminal can be an LU6.2 partner, but cannot be a TCP/IP partner

Figure 9-14. CICS to CICS Intercommunication Services CI207.0

Notes:

CICS-to-CICS intercommunication facilities supply support for:

- **Transaction Routing**, which allows an application to be run in a different CICS region from the one to which the terminal is connected. The terminal owner is called the TOR, while the application owning region is called the AOR. The TOR may act as a router to different AORs.
- **Distributed Program Link (DPL)**, which allows a link to be made from one program to another program running in a different CICS region. Both CICS regions are AORs. The front-end AOR may act as a router to different back-end AORs.
- **Function Shipping**, which allows an application to access a CICS resource in a different CICS region. The resources can be CICS queues (TD or TS) or VSAM files. Depending on the resource, CICS resource-owning regions may have names such as a FOR (for a file), or QOR (for a queue), or TSOR, or TDOR, or whatever...
- **Asynchronous Transaction Processing**, which allows an application to issue an asynchronous EXEC CICS START command to start a transaction in a different CICS region. Both CICS are AORs. The front-end AOR may act as a router to different AORs.

Transaction Routing

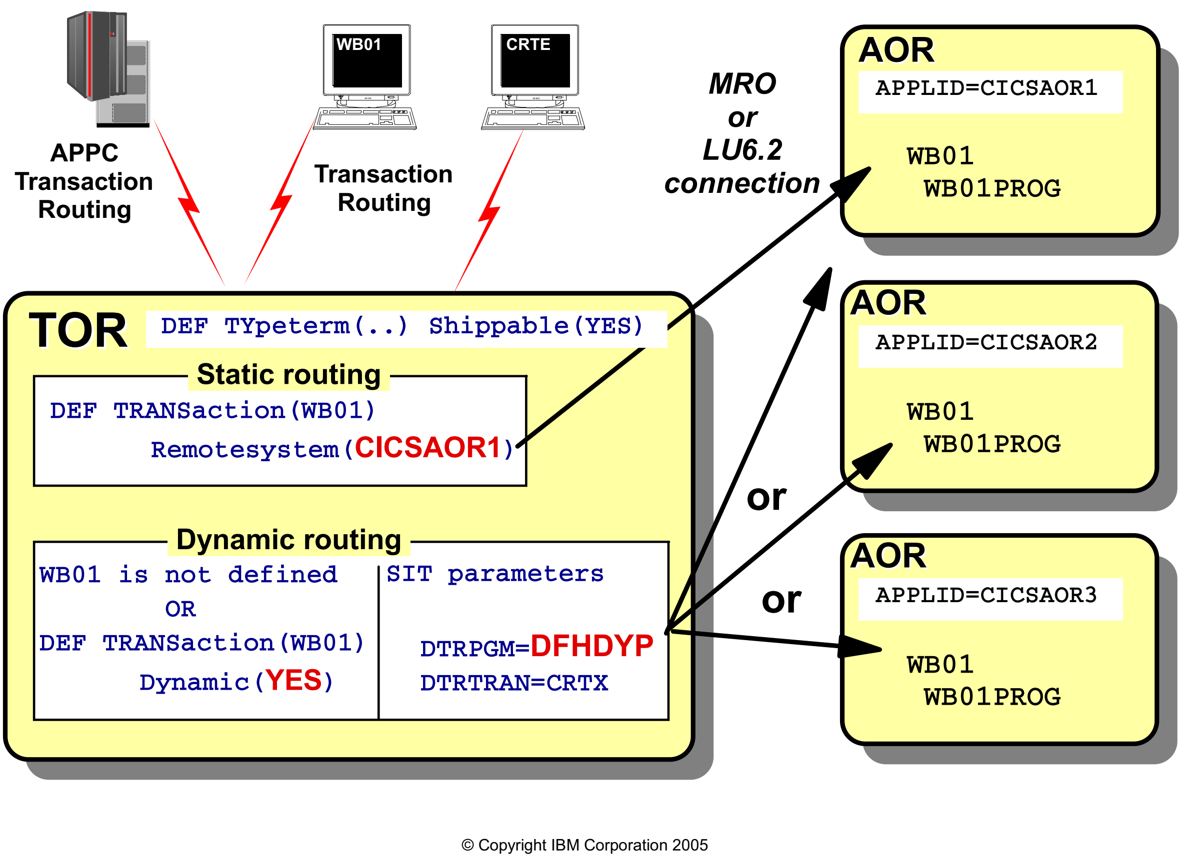


Figure 9-15. Transaction Routing CI207.0

Notes:

Transaction Routing of a transaction-initiation request issued from an APPC device or a terminal requires:

- An MRO or LU6.2 communication link between both CICS regions
- A REMOTE terminal definition in the AOR.
  If the terminal TYPETERM is defined with the SHIPPABLE(YES) attribute in the TOR, no definition is required within the AOR. The TOR will ship it along with the transaction initiation request.
- A REMOTE transaction definition in the TOR.
  If the transaction is defined with the DYNAMIC(NO) attribute, static routing will occur to the AOR named by the REMOTESYSTEM attribute of the transaction definition.
  If the transaction is *not* defined, or is defined with the DYNAMIC(YES) attribute, dynamic routing will occur. The program named by the DTRPGM SIT attribute is invoked and selects the AOR to route the transaction to. A routing decision may be made based on the workload each AOR is currently running.

Notice that the user is not aware that the transaction has been function-shipped.

The CRTE system transaction can be used to create a logical connection between a terminal connected to a TOR and an AOR. Once this connection is established, any transaction entered from the terminal is routed to the AOR. In other words, once connected to the TOR, a system programmer may run the CEMT transaction in any CICS region connected to this TOR, without the need to disconnect from the TOR and reconnect to the AOR.

Function Shipping

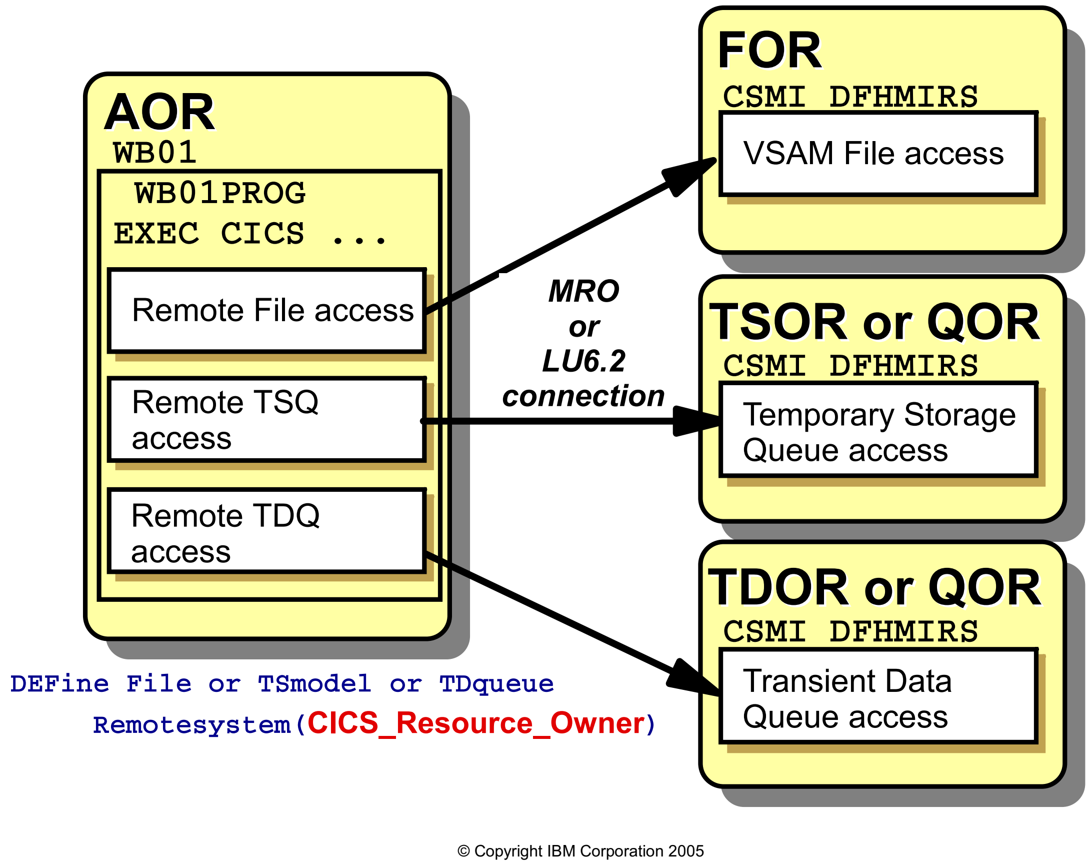


Figure 9-16. Function Shipping CI207.0

Notes:

Function Shipping enables CICS applications to access CICS files and queues that are owned by a remote CICS region.

The distribution is specified and performed at system level, without the application having to be aware of it.

- The data resource definition (file or queue) in the region where the application executes carries the *Remotesystem* name specification.
- The command to be executed is sent to the remote region with the special system transaction CSMI, the so-called **mirror transaction**.
- On the remote resource-owning side, the mirror transaction starts the system mirror program **DFHMIRS.** Within this program, the command shipped is executed, and its result is returned to the originating application.

By use of function shipping, files, temporary storage, and transient data queues may be accessed from within multiple CICS regions with full update integrity. The terms **File owning region** (FOR) and **Queue owning region** (QOR) are commonly used in this context.

Distributed Program Link (DPL)

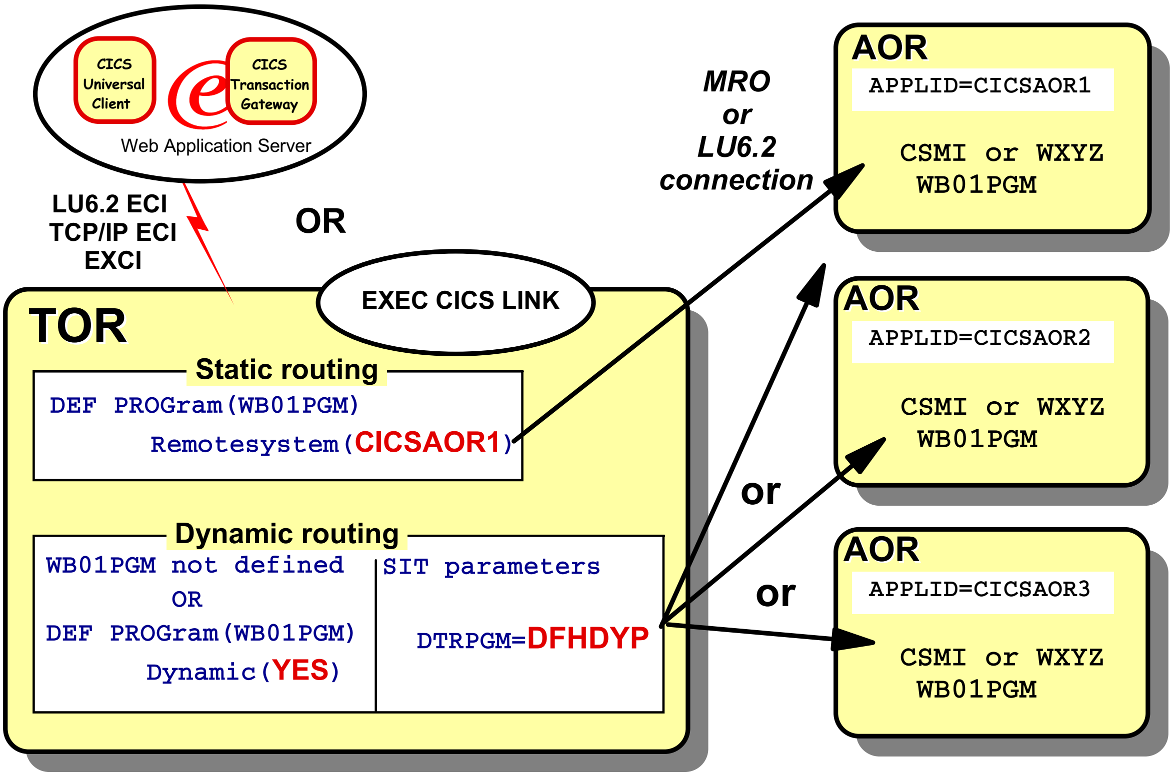


Figure 9-17. Distributed Program Link (DPL) CI207.0

Notes:

Distributed Program Link (DPL) allows a program executing in one CICS to link to a subprogram that executes in another CICS.

- "Client" and "Server" programs communicate via a communication area (COMMAREA), which is limited to 32 KBytes in size.
- The underlying system technique is the same as for function shipping, namely the mirror transaction and program.
  - If the server program is to run under a specific transaction code different from the system mirror transaction, this has to be coded in the following way:
    - *On the client side*, the server program definition must name this transaction code with the TRANSACTION parameter.
    - *On the server side*, this transaction must be defined as a local transaction, and specify the system mirror program, DFHMIRS, as its associated initial program.
- The server program must not use terminal-oriented commands, as there is no terminal environment available to the mirror transaction.

- So, the DPL is primarily intended to be used in a distributed environment, where the client program covers the presentation logic, which may be based on CICS or not. In the latter case, the CTG or CUC connector can be used to issue the link request in the form of an ECI call.

In a similar way to transaction routing, dynamic routing will occur if the program is not defined, or if it is defined with the DYNAMIC(YES) attribute. As for transaction routing, the DTRPGM SIT parameter names the dynamic routing program which is responsible for naming the target AOR.

Note, again, that DPL is absolutely transparent to both the user and the application.

Asynchronous Transaction Processing (ATP)

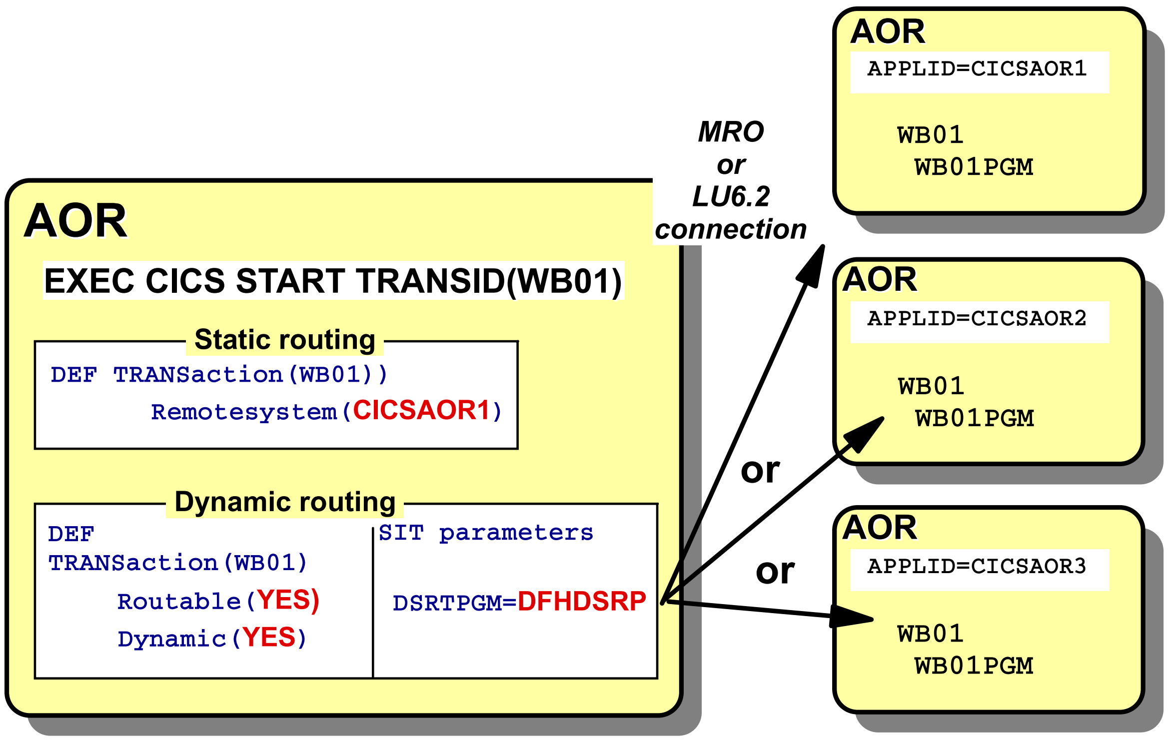


Figure 9-18. Asynchronous Transaction Processing (ATP) CI207.0

Notes:

Asynchronous Transaction Processing (ATP) has characteristics in common with both Transaction Routing and Function Shipping.

- ATP is the option to use an `EXEC CICS START` command to start a transaction from within an application. The started transaction may be defined as a remote transaction, just as for Transaction Routing.
- The started transaction is launched and run *asynchronously* to the originating application, in the sense that the latter is not informed about the successful execution, nor even about the successful start of the named transaction.
  So, the `EXEC CICS START TRANSID` command may be understood as an order to CICS to start the named transaction either as soon as possible or after a specified time (`INTERVAL` option specified), and CICS returns a normal response to this command, if the transaction and the region in which to run it is known to it.

In practice, further considerations have to be taken about whether the started transaction is associated to a terminal (`TERMID` option specified) or not.

Again, static or dynamic routing will occur depending on the DYNAMIC(YES) transaction attribute. As the DYNAMIC(YES) attribute is shared with transaction routing, the ROUTABLE(YES) transaction attribute will be used along with DYNAMIC(YES) for ATP. In order to keep us awake, the SIT parameter naming the routing program is not DTRPGM but DSRTPGM for ATP.

As you suspect, ATP is absolutely transparent to both the user and to the application.

Connecting Systems (Summary)

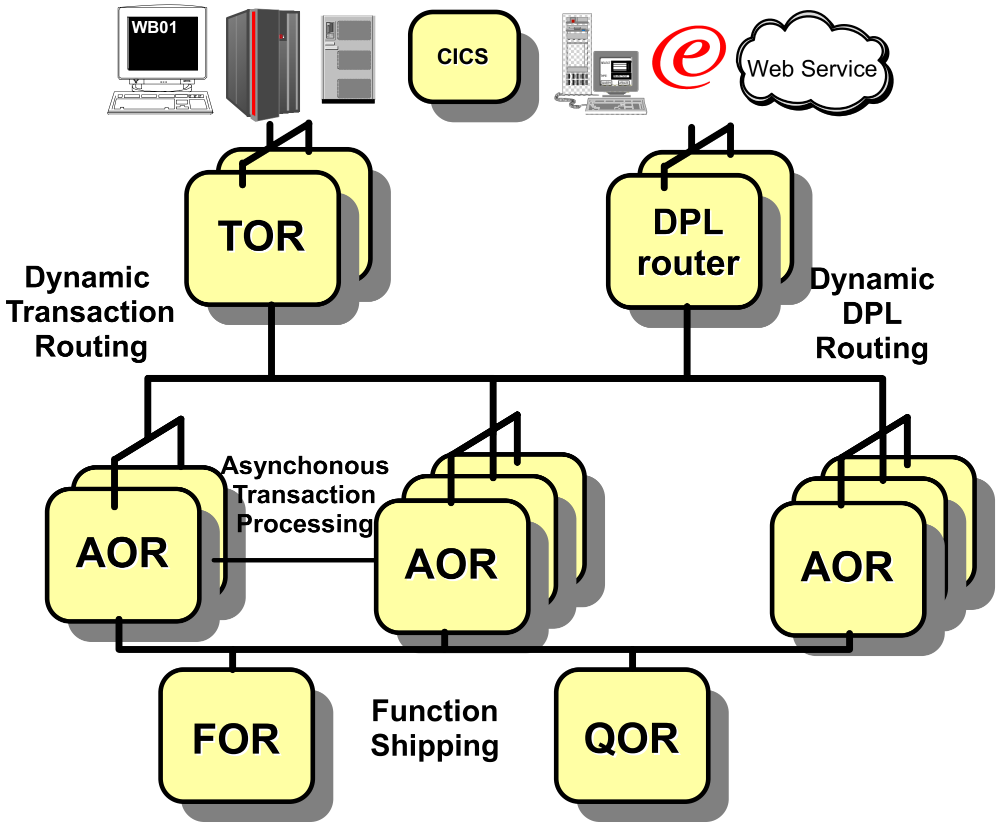


Figure 9-19. Connecting Systems (Summary) CI207.0

Notes:

This figure shows a typical CICS complex composed of three layers:

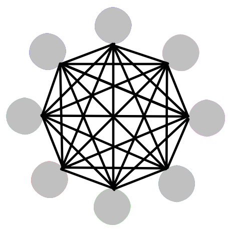
1. Listening CICS regions:
   This can be a set of SNA TORs sharing the same VTAM generic network name or not. Typically these TORs use the dynamic transaction routing function to distribute the workload toward a set of (possibly cloned) AORs.
   This can be a set of TCP/IP listener regions typically handling HTTP and ECI requests. These regions use the dynamic DPL function to distribute the workload toward a set of (possibly cloned) AORs.
2. Application-owning regions:
   They run the business logic and use the EXEC CICS APIs to access resources. They may use the ATP function to start CICS processes in other CICS regions asynchronously. They typically use the function shipping function to access shared queues and files.
3. Resource owning regions:
   They run mirror transactions to access queues and files shared by a set of AORs.

9.5 CICS SOAP Support

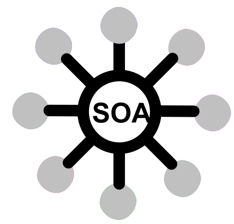
Service Oriented Architecture

A system architecture in which application functions are built as components (or "services") that are **loosely-coupled** and **well-defined** to support interoperability, and to improve flexibility and reuse

- **Loosely-coupled:** the underlying implementation is hidden from the application that invokes the service
- **Well-defined:** common definition of services which is independent of any particular technology but can be used by all technologies



Traditional integration involves **technology-aware bridges** between app components, thus is **complex and expensive** to maintain



SOA is a **common way of interacting** that can be used with any component, **regardless of the underlying technology**, greatly **simplifying and strengthening integration efforts**

WEB SERVICEs is an implementation of an SOA

Figure 9-20. Service Oriented Architecture CI207.0

Notes:

IT systems were originally built to facilitate manual work. The trend is increasing automation and integration on a larger scale: computers were introduced to speed the work, but lack of interoperability between systems starts to become a problem.

As business continues to grow and becomes more complex, there is a need to deal quickly and efficiently with challenges like new business opportunities, new customer requirements, or competitive threats.

End-to-end integration, internally as well as between partners, is a strong requirement. Integration needs to be flexible, and independent of the technology choices for each component in the integrated system.

SOA addresses several issues:

- Interoperability issues between applications in different programming languages, on different hardware or operating systems, from different vendors.
- Flexibility issues by implementing universal interfaces that can cope with inevitable changes in software caused by changing business needs.

- Implementation dependencies issues: an SOA must be based on open, vendor- and platform-neutral standards. Java would be the solution if everyone used it, but some systems use other technologies.
- Existing investment issues, so that old software can to be used in new ways by building a Web services layer for universal access.
- Distributed computing programming issues, so that it is as easy as local programming.

And more!

Web Services and SOAP

SOAP is an extensible XML protocol to invoke a "function" on a server to perform a given operation. It allows the following to be defined:
- Types of exchanges between service requesters and service providers
- Messages exchanged by service requesters and service providers

WSDL is an XML vocabulary to describe **abstract** Web Services interfaces:
- Service definition (name, location, transport protocol)
- Operations definitions (name, input and output messages)
- Messages and data types definitions (message structures, data types)

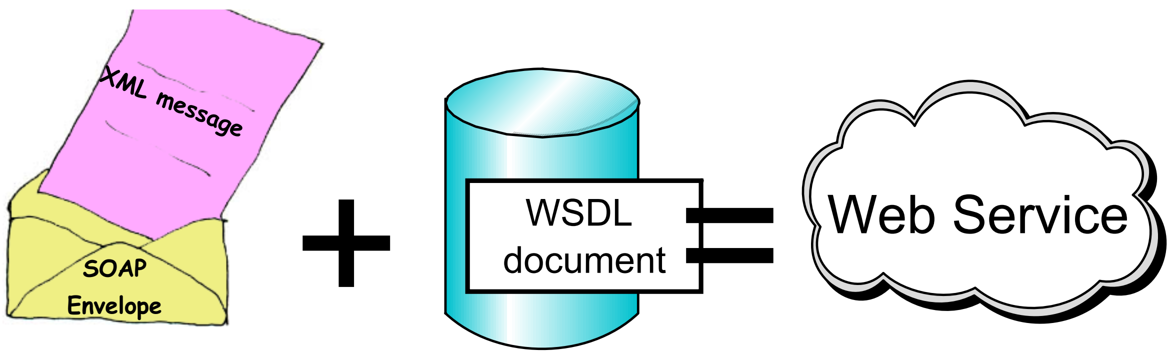


Figure 9-21. Web Services and SOAP CI207.0

Notes:

What is a **software service**?

It is a set of application functions that show the components of a business process. These functions are:

- Componentized
  Meaning that they use encapsulated code designed for reuse
- Loosely-coupled
  Meaning that they use a well-defined interface designed to show business functions and data, but also to hide underlying implementation details from service requesters.
- Stateless
  Meaning that services are not dependent on the context or state of other services.

A service either provides information, or facilitates a change to business data from one valid and consistent state to another. They should be invocable through defined communication protocols that stress interoperability and location transparency.

What is a **Web Service**?

The Web Services Architecture draft provides a Web Service definition:

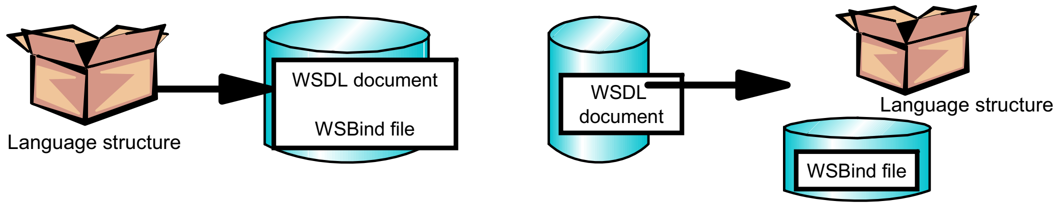
A Web service is a software system identified by a URI, whose public interfaces and bindings are defined and described using XML. Its definition can be discovered by other software systems. These systems may then interact with the Web service in a manner prescribed by its definition, using XML-based messages conveyed by Internet protocols.

Current **Web Service** implementation definition:

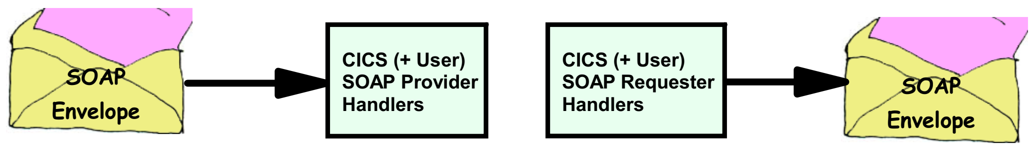
Web services are software components described via WSDL XML documents which are capable of being accessed via standard network protocols such as SOAP over HTTP.

CICS and SOAP

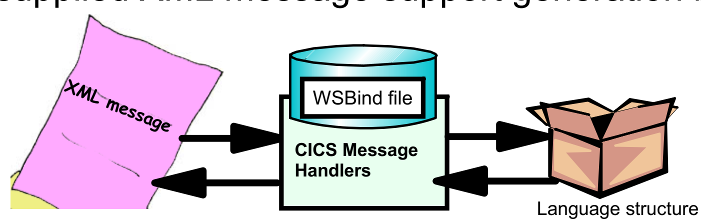
- CICS-supplied WSDL and XML data mapping batch utilities



- SOAP protocol handling



- CICS-supplied XML message support generation batch utilities



- CICS-supplied API commands

Figure 9-22. CICS and SOAP CI207.0

Notes:

CICS TS V3.1 supplies extensive support for SOAP-based Web Services.

- A batch utility to generate a WSDL service definition document from a language structure definition, typically a COMMAREA. You may also generate the WSDL document from a language structure definition using the WebSphere Studio Enterprise Developer tooling.
  This utility also generates a WSBIND file which will be used by CICS SOAP handlers to map the Language Structure elements to the XML message elements. This eliminates the need for programmers to code XML handling code.
- A batch utility to generate a language structure definition from a WSDL service definition document.
  This utility also generates a WSBIND file which will be used by CICS SOAP handlers to map the Language Structure elements to the XML message elements. This eliminates the need for programmers to code XML handling code.
- Default SOAP protocol handlers which may be complemented by private handlers dedicated to private SOAP headers.

Quality of service is implemented through standard SOAP headers. CICS supports the WS-SECURITY and WS-ATOMICTRANSACTION specifications.

- XML message handling is supported by CICS handlers which use the WSBIND file information generated by the batch utilities.
  XML messages can also be handled by private handlers using the compiler-supplied XML parse or generation facilities, or both.
  WebSphere Studio Enterprise Developer tooling can also be used to create XML message handlers from a language structure.

CICS supplies Web Services-dedicated API commands, such as EXEC CICS INVOKE WEBSERVICE or EXEC CICS SOAPFAULT CREATE.

SOAP Protocol

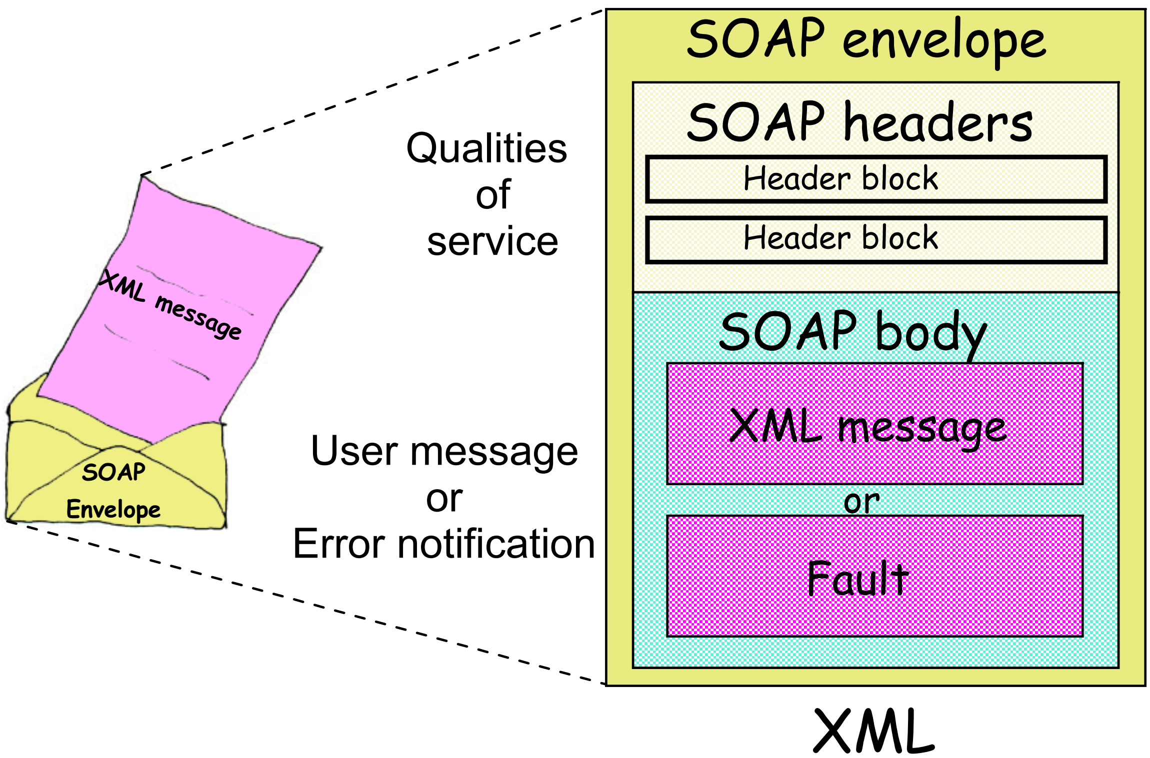


Figure 9-23. SOAP Protocol CI207.0

Notes:

The SOAP protocol is based on XML. Any message must be composed of the following XML elements:

- The mandatory SOAP ENVELOPE element
- Zero or more optional SOAP HEADERs element.
  Headers are the way to adapt the SOAP protocol to Web Services extensions such as WS-Security or WS-Atomictransaction. SOAP also supplies its own headers in order to indicate the target system URL of a service provider.
- The mandatory SOAP BODY element.
  The SOAP body contains the user application XML message *or* a SOAP error notification in the form of the SOAP FAULT element which is used by Web Services applications to notify each other of an error.

CICS Web Service Resource Definitions

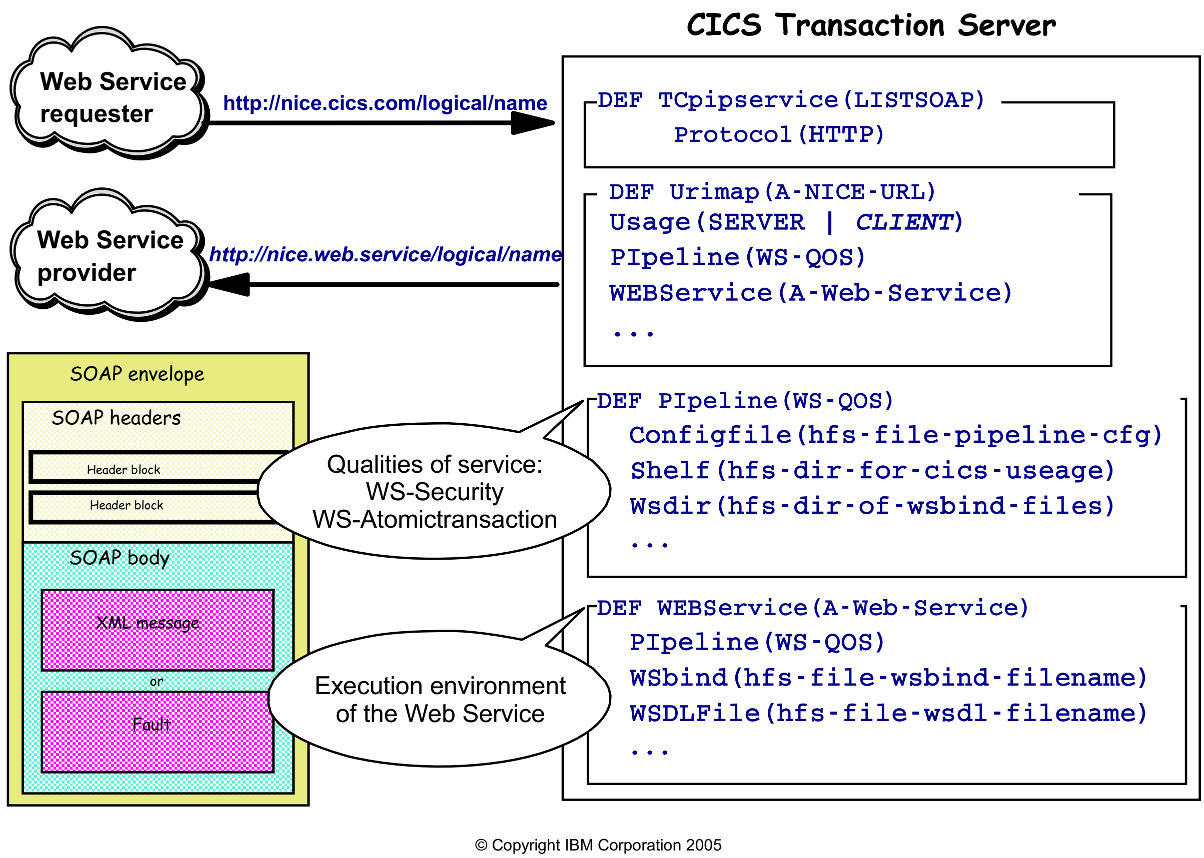


Figure 9-24. CICS Web Service Resource Definitions CI207.0

Notes:

The figure shows the CICS resource definitions required to support Web Services using the SOAP over HTTP binding. CICS also supports SOAP over WebSphere MQSeries; in such a case, the TCPIPSERVICE definition is not required.

The SOAP protocol is like Russian dolls: the XML message is in the SOAP body, which is in the SOAP envelope, which is in the HTTP body and which may contain SOAP headers.

CICS supplies its own handlers for each part of a SOAP message. It also allows for private handlers. The names and sequence of SOAP handlers are defined in an XML configuration file stored in an hfs. A set of handlers is seen as a CICS PIPELINE resource. The PIPELINE definition supplies the XML configuration hfs file name along with the hfs directory name (note that the file as a pipeline may handle multiple messages) of the WSBIND files generated by the CICS SOAP utilities.

When a SOAP request is processed, the pipeline definition is referenced by the URIMAP definition corresponding to an inbound (Web Service requester to CICS) or outbound (CICS to Web Service provider) request.

The WEBSERVICE definition supplies the WSBIND file name (which is within the pipeline-specified hfs directory) and the WSDL file, if any, which describes the Web Service interfaces and bindings. If the WSDL file is available, the WEBSERVICE definition may request an optional request validation against it before processing.

Web Service Batch Utilities

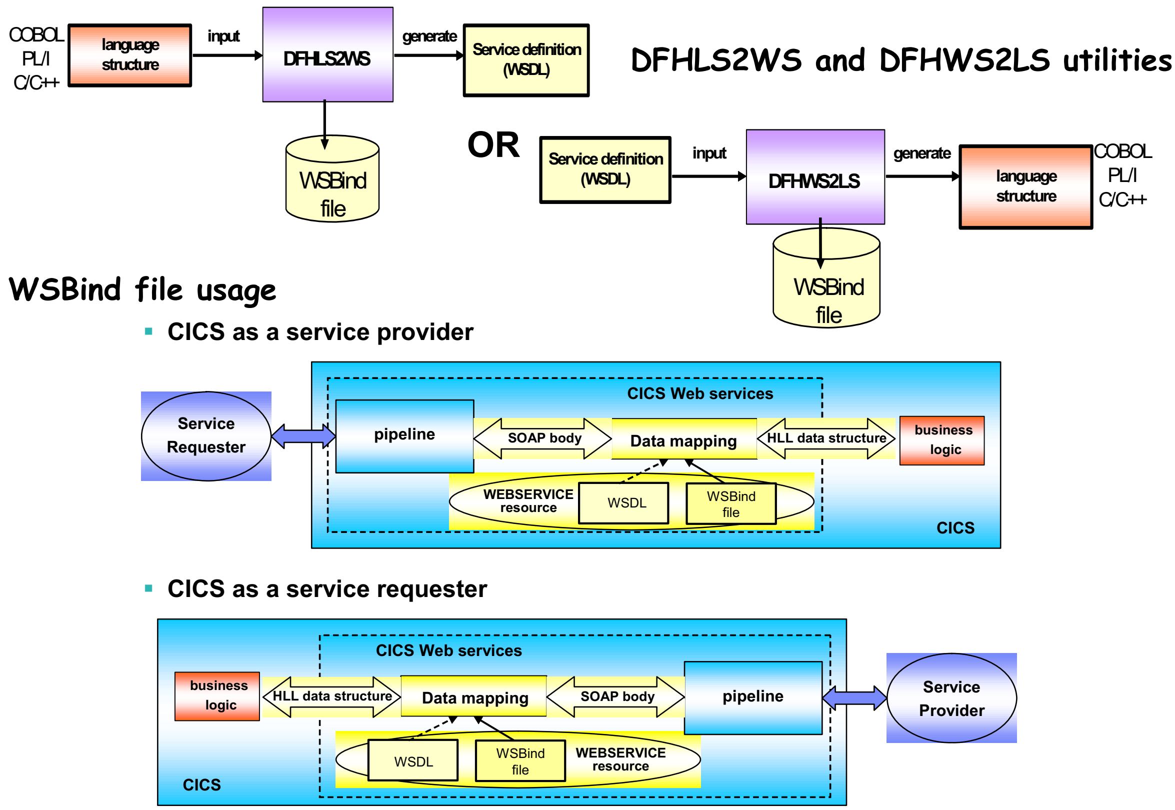


Figure 9-25. Web Service Batch Utilities CI207.0

Notes:

CICS supplies two batch utilities dedicated to the Web Services support:

- DFHLS2WS and DFHWS2LS

DFHLS2WS generates the WSDL file describing a CICS Web Service interface from a language structure. This is typically used when presenting an existing CICS application as a Web Service provider.

DFHWS2LS generates a language structure from the user XML message described by a WSDL definition file. This is typically used when accessing an existing Web Service provider from a CICS Web Service requester application.

Both utilities generate a WSBIND file used to map an XML message to a language structure as shown in the second part of the figure.

The data mapping code is CICS code and does not require any user application coding.

Unit Summary

Having completed this unit, you should be able to:

- Provide definitions to support the connection of LU6.2 devices
- Configure the LU6.2 support of the CICS terminal autoinstall function
- Provide definitions to support connection of TCP/IP partner systems using the HTTP protocol, the IIOP protocol or the ECI protocol
- Provide definitions to support MRO connections between CICS systems
- Provide definitions to support EXCI connections from batch programs
- Describe the dynamic routing facilities provided by CICS
- Provide definitions required to access remote CICS resources
- Provide definitions to support connection of Web Services requesters using the SOAP protocol
- Provide definitions to support connection to Web Services providers using the SOAP protocol

Figure 9-26. Unit Summary CI207.0

Notes:

Unit 10. System Control

What This Unit Is About

This unit introduces the basic system control facilities of CICS TS and the parameters available to control these.

What You Should Be Able to Do

After completing this unit, you should be able to:

- State the use of the task parameter MXT, and describe how this and other values may be altered with CEMT and application programs
- State the use and definition of transaction classes
- Discuss the implementation of task priority
- Control the time an application waits for a resource another user owns, or user think time

References

CICS Transaction Server for z/OS Version 3.1 Installation Guide, GC34-6426

CICS System Definition Guide, SC34-6428

CICS Supplied Transactions, SC34-6432

CICS System Programming Reference, SC34-6435

Unit Objectives

After completing this unit, you should be able to:

- State the use of the task parameter MXT, and describe how this and other values may be altered with CEMT and application programs
- State the use and definition of transaction classes
- Discuss the implementation of task priority
- Control the time an application waits for a resource another user owns, or user think time

Figure 10-1. Unit Objectives CI207.0

Notes:

10.1 Task Dispatching and Transaction Management

Multitasking

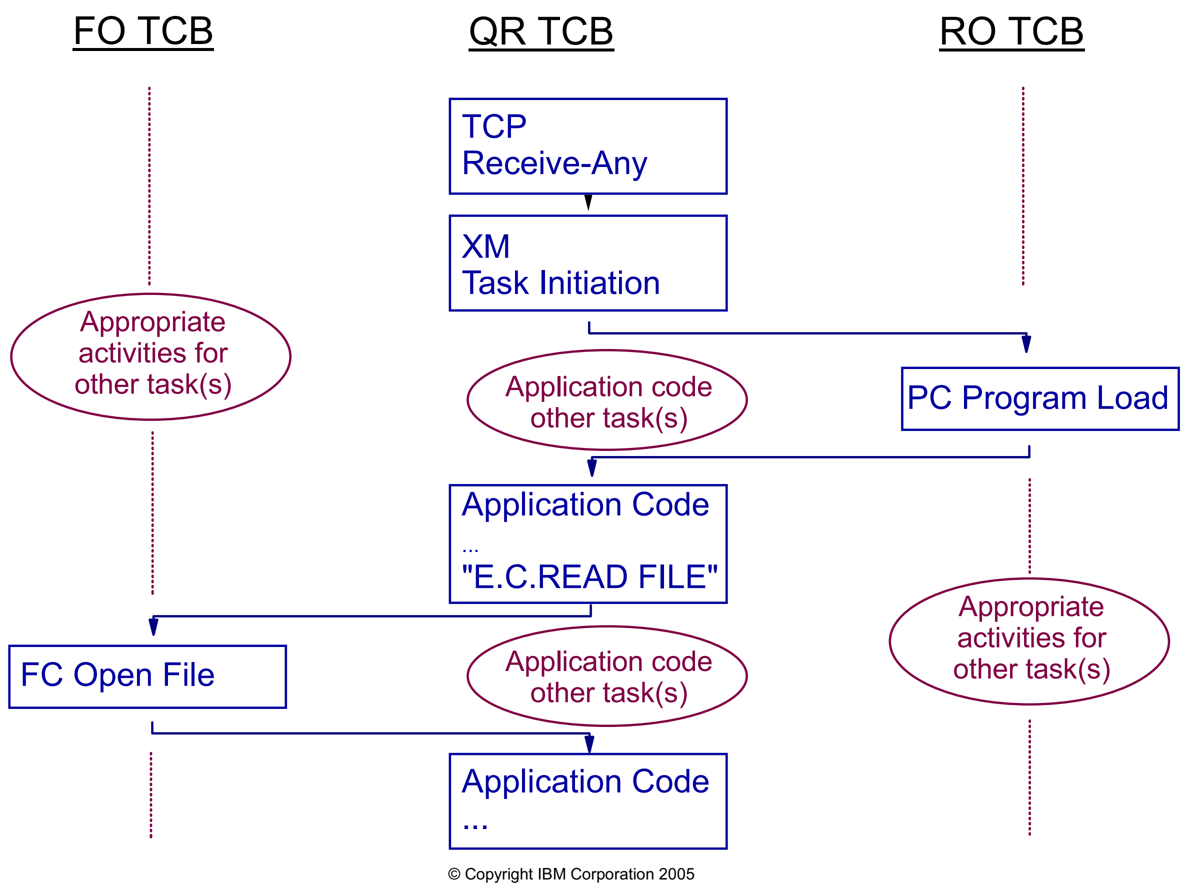


Figure 10-2. Multitasking CI207.0

Notes:

- A dispatcher “default task” is active for each TCB, giving control to the CICS task with the highest priority that is ready to run under the respective TCB
  - **QR TCB Quasi-reentrant**
    All application code and CICS code that is non-reentrant.
    The QR TCB attaches a number of subtask TCBs for special purposes, for example opening and closing of data sets.
  - **RO TCB Resource owning**
    CICS code containing imbedded MVS WAIT, for example program load.
  - **FO TCB File owning TCB**
    Used for opening and closing user data sets.
- **Support for Open Transaction Environment** enables specified user tasks to run under their own task control block (TCB). The initial beneficiaries of this change are Java application programs that run under a Java virtual machine (JVM), starting with CICS TS 2.2, the DB2 external resource manager interface, and, with CICS TS 3.1, the tasks that invoke programs enabled with OPENAPI option.

Threadsafe and Openapi

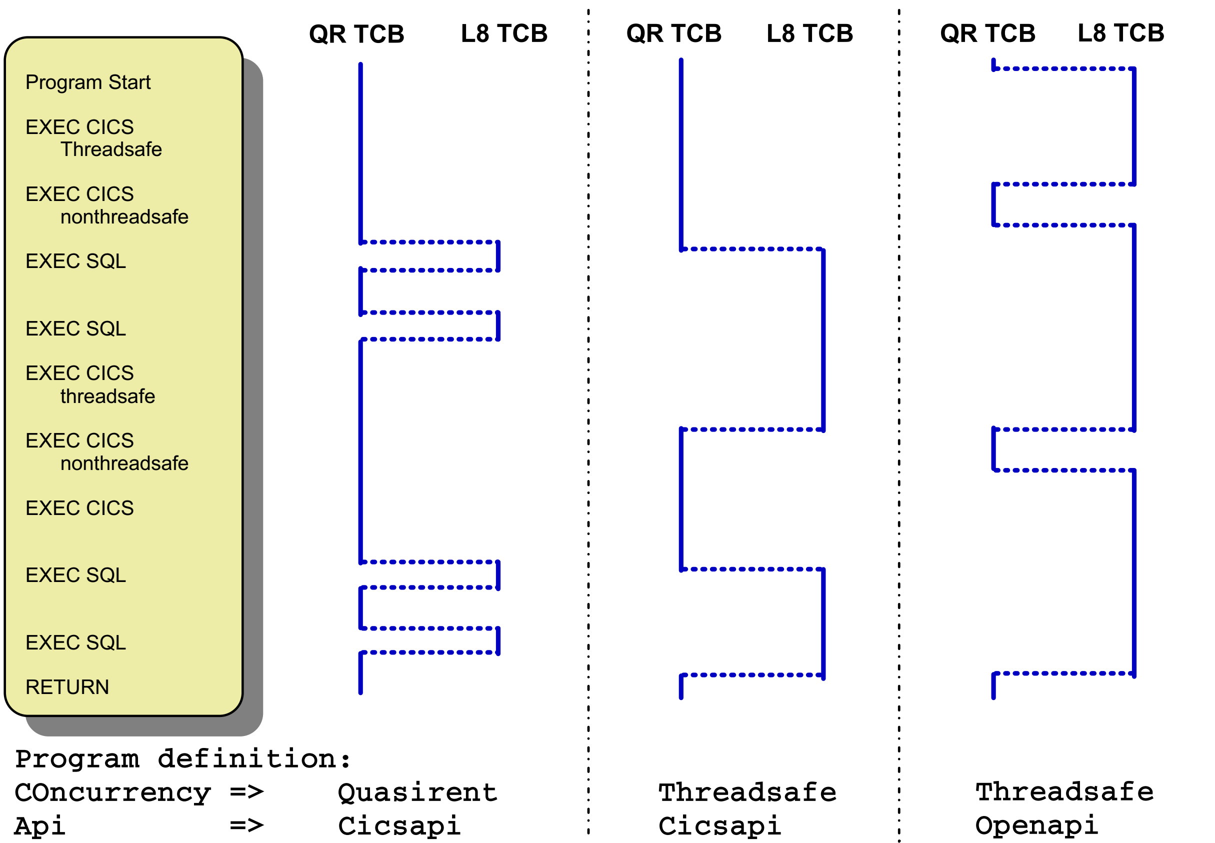


Figure 10-3. Threadsafe and Openapi CI207.0

Notes:

For programs written in languages other than Java, the **Concurrency** and **Api** parameter definitions determine the TCB under which the task will run.

The main benefit of this support is that it allows application workloads to be moved from a single QR TCB onto multiple TCBs, enabling better utilization of machine resources to achieve better throughput.

- When a program that is not defined as **threadsafe** makes a DB2 request, CICS switches from the QR TCB (where the program is executing) to an open TCB, and back to the QR TCB again when the DB2 request is complete.
- With program-defined **threadsafe** and **cicsapi**, CICS switches from the QR TCB to an open TCB upon the first DB2 request. Programs execute under this open TCB until a nonthreadsafe CICS command is issued. At this time, CICS switches to the QR TCB. Programs execute under this TCB until the next SQL command is issued.

- When a program that is defined **threadsafe** and **openapi** invokes a nonthreadsafe CICS command, CICS switches from the open TCB to the QR TCB and back to an open TCB when the command is complete.

The commands that are threadsafe are indicated in the command syntax diagrams in the *CICS Application Programming Reference* and the *CICS System Programming Reference* with the statement **This command is threadsafe**, and are listed in a topic of the *CICS Application Programming Reference, SC34-6434* and in Appendix D of the *CICS System Programming Reference, SC34-6435*.

Task Parameters

SIT

.
.
MXT=15
.
.
.

CEMT INQ [SYSTEM] MAXTASKS

CEMT SET MAXTASKS(20)

CEDA DEF TRANCLASS

TRANClass ==> CLASSxy

MAXACTIVE ==> 5

PURGETHRESH ==> No| nn

CEDA DEF TRANSACTION

TRANSaction ==> ABC1
.
.
TRANClass ==> CLASSxy

Figure 10-4. Task Parameters CI207.0

Notes:

- Maximum tasks (MXT) is the maximum number of user tasks that will be accepted into the system.
  The range is 1 through 999; the default is 5.
  MXT does not include system tasks.

  User tasks above this number are queued by CICS but not processed (attached) until the number of user tasks attached drops below the MXT limit.

- The TRANCLASS resource type allows grouping transactions into classes, in order to:
  - Limit the number of transactions that are active at a time within that class (MAXACTIVE)
  - Determine the handling of transactions entered above the specified limit (PURGETHRESH).

- IBM-supplied RDO group DFHTCL contains definitions for Tranclass DFHTCL01 to 10 that are assigned to the old Transaction classes 1 to 10 for compatibility with pre-Version 4 transaction definitions.

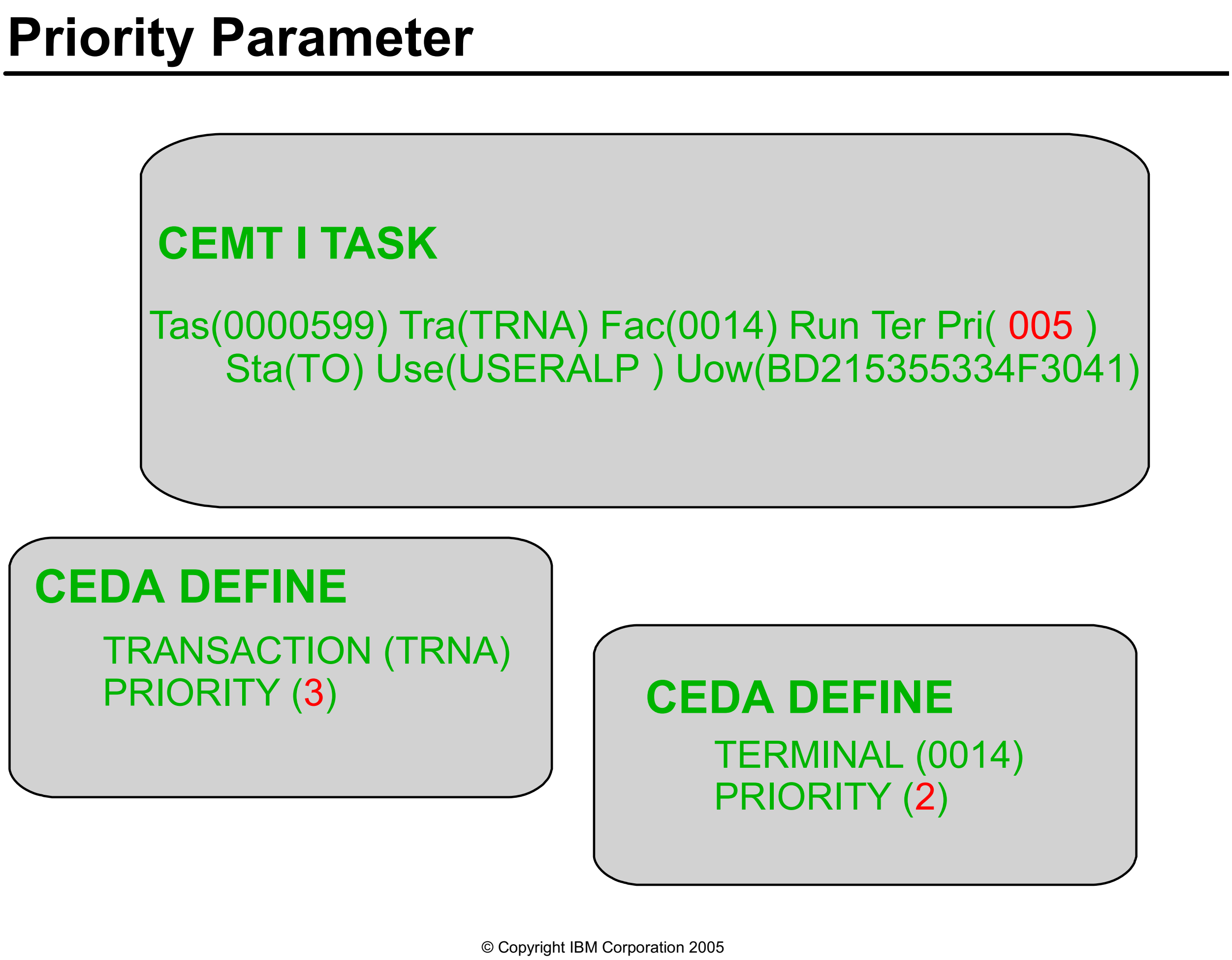


Figure 10-5. Priority Parameter CI207.0

Notes:

- Priority is low to high (1-255), with 1 being the default for transaction, and 0 for terminal.
- Transaction and terminal priorities are set at define time.
- Task execution priority is determined by adding the transaction, terminal, and user priorities.
- Task, transaction, or terminal priorities may be altered by CEMT or by SPI.

Time Parameters

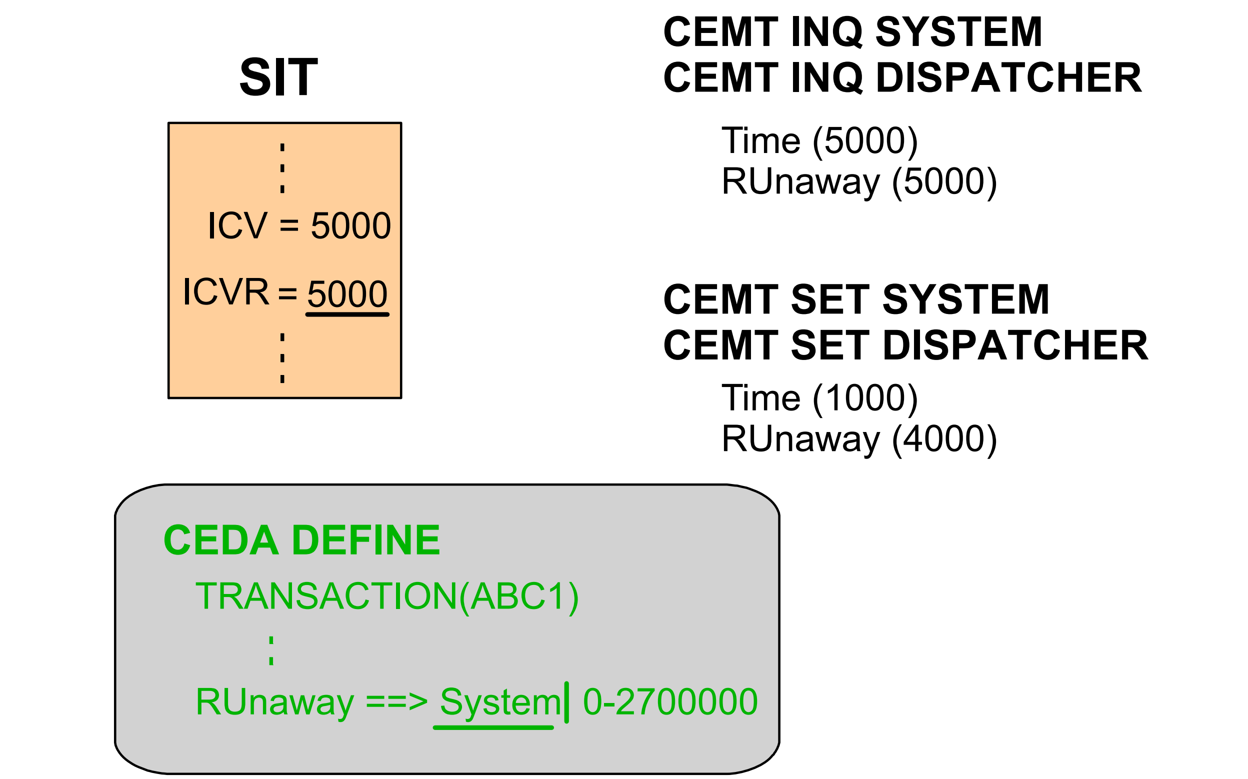


Figure 10-6. Time Parameters CI207.0

Notes:

- **ICV** is the maximum time CICS will release control to the operating system when CICS has no work to perform.
  - Any interrupt, for example data set I/O or terminal I/O, will dispatch CICS before the interval expires.
  - A low value for ICV can impact batch performance by frequent dispatches of CICS when there is no work to perform. A high value can delay processing time-dependent events.
  - The default is 1000 (milliseconds). The recommendation is to start at 5000, and reduce if the value appears too high.
- In CICS terms, **runaway task time** is the task CPU time between EXEC CICS commands that CICS will allow a task to accumulate before considering it a “runaway” (looping) task and abending it with abend code **AICA**.
  - This time may be specified individually within the transaction definition by use of the `Runaway` parameter. If not, the **ICVR** value specified in the SIT applies.

  - To optimize, monitor your applications and code the limit appropriately, allowing for a factor of safety.

- These values may be inquired on, and changed during CICS operation, using the CEMT options shown.
- The *CICS Performance Guide, SC34-6452*, contains a detailed discussion of these values.
- The **`CEMT INQUIRE/SET DISPATCHER`** command is introduced in CICS TS for z/OS V2. In addition to the values that are also accessible through the INQUIRE/SET SYSTEM command, it returns information about the number of the active OTE TCBs mentioned beforehand, and allows you to change these specifications online.

Task Timeout Parameters

CEDA ALTER/DEFINE
TRANSACTION(UPDT)

PROFILE (OPTIME)
:
SPURGE (YES)
DTIMOUT (0030)
RUNAWAY (2000)

CEDA ALTER/DEFINE PROFILE(OPTIME)
:
RTIMOUT (1000)

Figure 10-7. Task Timeout Parameters CI207.0

Notes:

- The *CICS Resource Definition Guide, SC34-6430*, contains information for defining transactions and profiles.
- **Deadlock time out** (`DTIMOUT`) sets the interval of time before a waiting task will be terminated. The default is no, which means that no deadlock detection will be performed.
  - The deadlock timeout will be honored only if the system purge (`SPURGE`) is coded as YES. Otherwise, the task continues to wait.
  - The task abend code is **AKCS**
- Terminal-read time out (`RTIMOUT`) sets the time interval that conversational tasks will wait for a response from an outstanding terminal read.
  - The task abend code is **AKCT**
- Time is expressed in minutes and seconds (mmss).
- Use the recommended values, monitor abends, and adjust as necessary.

Purging Tasks -KILL Function and CEKL

- **KILL** option of CEMT and SPI commands: purge problem tasks, which may not be purgeable with PURGE or FORCEPURGE option.
  - SET TASK
  - SET CONNECTION
  - SET TERMINAL
- **CEKL** transaction: remove tasks from a system in a situation where you cannot issue the CEMT transaction.
  - Invoked from MVS console
  - Runs in its own TCB within CICS
  - Inquire support
  - Audit trail provided

Figure 10-8. Purging Tasks -KILL Function and CEKL CI207.0

Notes:

The KILL function is introduced, allowing looping or suspended tasks to be removed from a CICS region without the need to cancel and restart a region, enabling the system to be kept running during critical periods.

The addition of a KILL option to the CEMT commands and the EXEC CICS equivalents shown allows tasks to be canceled where that could not have been achieved using the PURGE or FORCEPURGE options.
This can also be invoked from CICSPlex SM.

The **CEKL** transaction can be invoked *from a console*, and runs on its own Task Control Block (TCB) within CICS. This means it can be used even in the event of a lockout of the CICS Quasi-Reentrant (QR) TCB.

`CEKL INQUIRE TASK` Lists information about tasks in the CICS region

`CEKL SET TASK PURGE|FORCEPURGE|KILL`
Cancels a selected task

An audit trail is provided for transactions that have been subjected to KILL.

System Control (Summary)

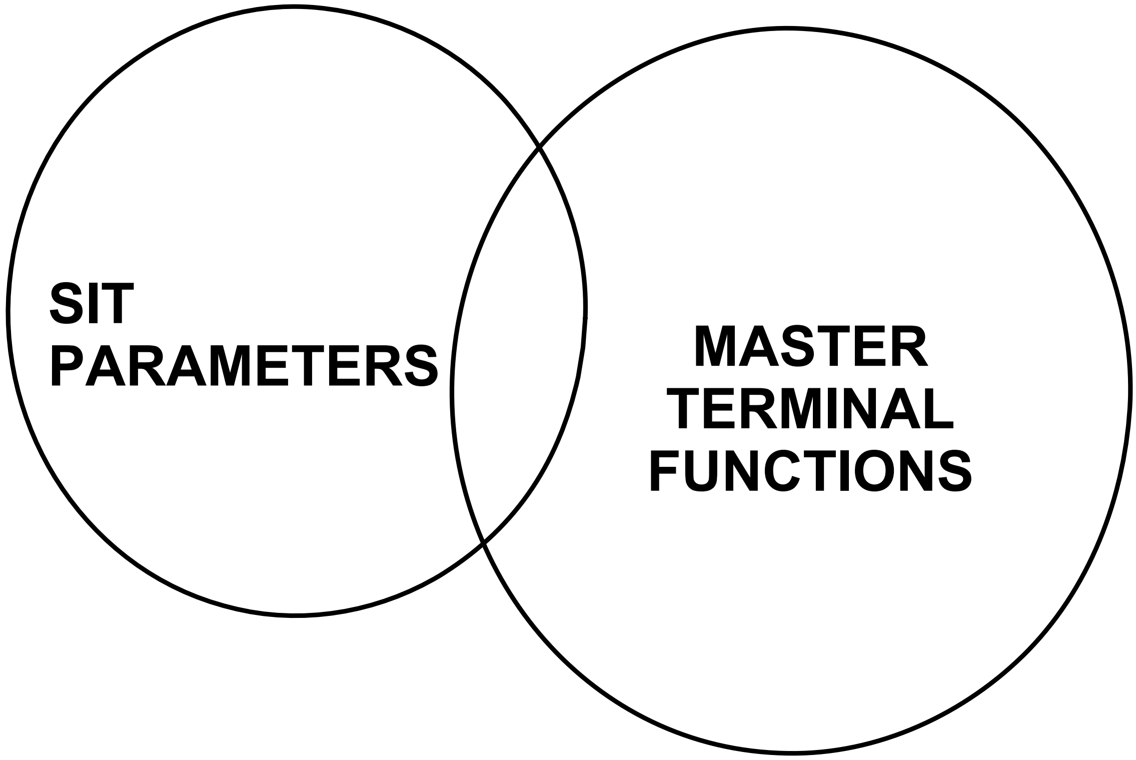


Figure 10-9. System Control (Summary) CI207.0

Notes:

- MXT sets the maximum number of user tasks CICS will initiate. It controls the virtual storage required by CICS TS.
- You may allocate transactions to certain transaction classes which have to be defined as RDO resources, and limit the number of active tasks within each transaction class.
- CICS TS offers a KILL option for use with CEMT SET commands and a console transaction CEKL, in order to terminate looping or “hanging” tasks.
- DTIMOUT is used to limit the time an application waits for a resource another user owns.
- RTIMOUT applies only to conversational tasks and limits the time for a user to reply.
- Many of the values may be altered with the CEMT transaction.
- Application programs may inquire and set values, eliminating the necessity for operator actions.

Unit Summary

Having completed this unit, you should be able to:

- State the use of the task parameter MXT, and describe how this and other values may be altered with CEMT and application programs
- State the use and definition of transaction classes
- Discuss the implementation of task priority
- Control the time an application waits for a resource another user owns, or user think time

Figure 10-10. Unit Summary CI207.0

Notes:

Unit 11. Storage Management and Storage Protection

What This Unit Is About

CICS manages the Dynamic Storage Area to provide storage needed for tasks and programs. This unit introduces you to the organization of a CICS region and the means and functions that are provided to avoid storage violations.

Furthermore, you will learn the considerations for making modules resident in the Link Pack Area or the DSA.

What You Should Be Able to Do

After completing this unit, you should be able to:

- List the main areas of the CICS address space
- State the main variables affecting CICS region size
- State the benefits of using the Link Pack Area, and define CICS and user programs as resident in the LPA
- Define region residency options for application programs and map sets
- Describe the features that protect CICS DSA storage against storage violations available in CICS TS for z/OS
- Activate storage protection and transaction isolation by use of SIT and resource definition parameters

References

CICS System Definition Guide, SC34-6428

CICS Resource Definition Guide, SC34-6430

Unit Objectives

After completing this unit, you should be able to:

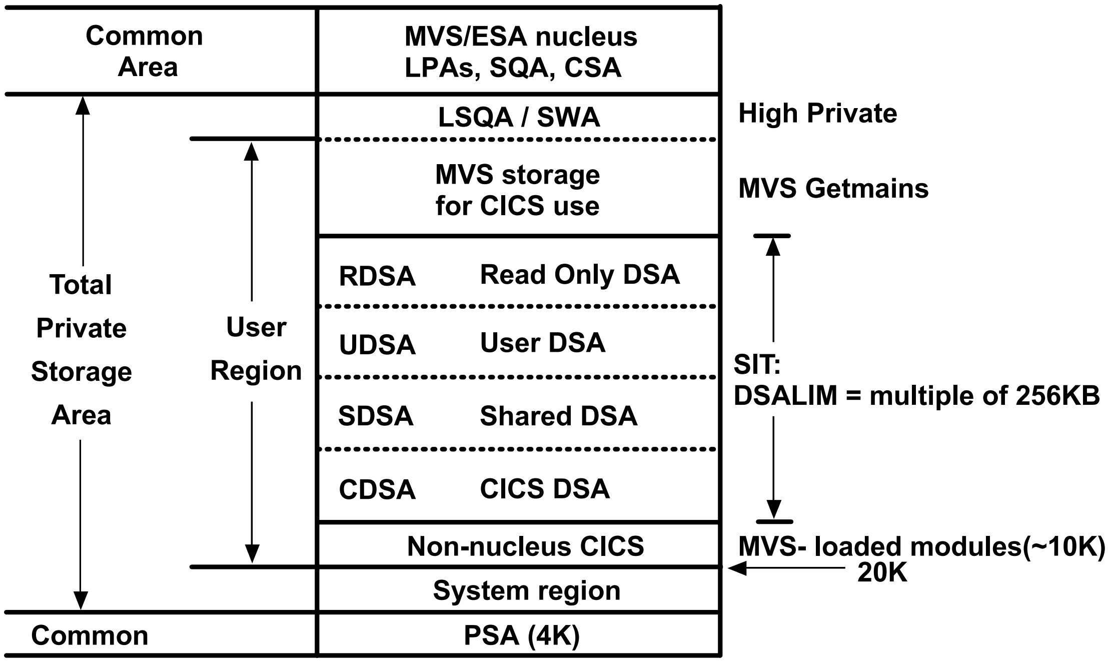
- List the main areas of the CICS address space
- State the main variables affecting CICS region size
- State the benefits of using the Link Pack Area, and define CICS and user programs as resident in the LPA
- Define region residency options for application programs and map sets
- Describe the features that protect CICS DSA storage against storage violations available in CICS TS for z/OS
- Activate storage protection and transaction isolation by use of SIT and resource definition parameters

Figure 11-1. Unit Objectives CI207.0

Notes:

11.1 Storage Management

CICS TS Storage Layout Below 16 MB



CSA	Common Storage Area	PSA	Prefixed Save Area
LPA	Link Pack Area	SQA	System Queue Area
LSQA	Local System Queue Area	SWA	Scheduler Work Area

Figure 11-2. CICS TS Storage Layout Below 16 MB CI207.0

Notes:

The diagrams on these pages show the storage areas of a CICS TS address space.

Common Area Shared by all address spaces. Size = n * 1024K

Private Area Exclusive to one address space. Size = n * 1024K

User Region Specified in JCL: REGION= (if < 16 M).

Within a CICS region:

- The **DSAs** include storage managed by CICS for programs and transactions, plus the CICS nucleus, which is loaded in the CDSA.
- The **MVS Free Storage**, left after allocating the DSA and Kernel Stack, is used to support MVS GETMAINs while CICS is running.
- CICS always manages the four DSAs shown, regardless of whether or not you use the features for which they have been designed.
  - **RDSA**: The key-0 storage area for all reentrant programs and tables below the 16MB boundary.

- **UDSA**: The user-key storage area for all user-key task-lifetime storage below the 16MB boundary.
- **SDSA**: The user-key storage area for any non-reentrant user-key RMODE(24) programs, and also for any storage obtained by programs issuing CICS GETMAIN commands for storage below the 16MB boundary with the shared option.
- **CDSA**: The CICS-key storage area for all non-reentrant CICS-key RMODE(24) programs, all CICS-key task-lifetime storage below the 16MB boundary, and for CICS control blocks that reside below the 16MB boundary.

• CICS dynamically manages the individual DSA automatically, but you may specify fix sizes for one or more DSAs by separate SIT overrides.

CICS TS Storage Layout Above 16 MB

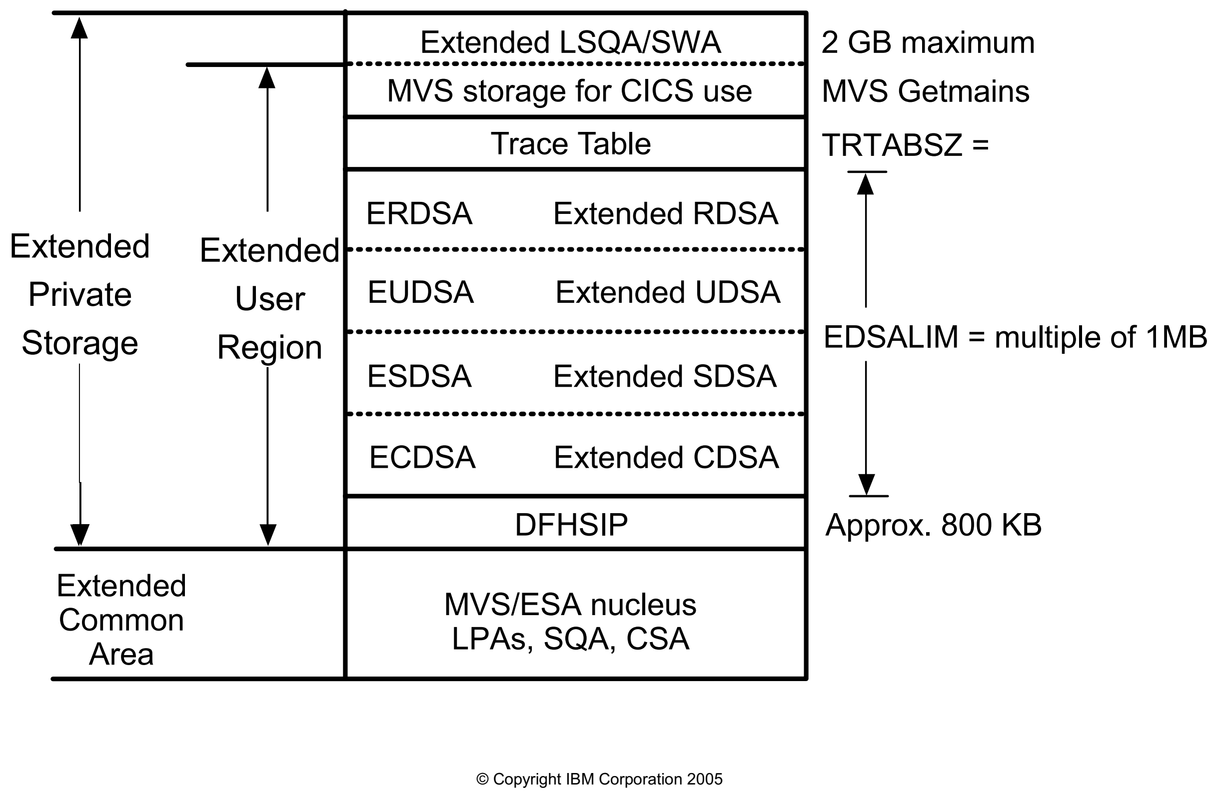
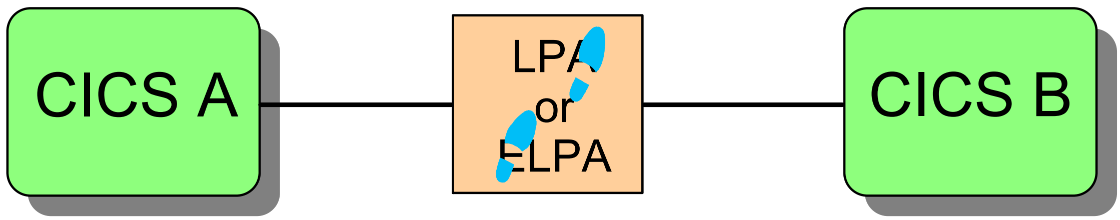


Figure 11-3. CICS TS Storage Layout Above 16 MB CI207.0

Notes:

- These areas are used in the same way as their counterparts below the line, and their individual sizes may be managed automatically by CICS or fixed by SIT overrides.
- The region size above the 16M can be controlled by specifying a region size greater than 16M. The default region size can be established by the SMF exit, IEFUSI. This default can be overridden by the `REGION=` value in JCL.
- If CICS runs out of storage, a console message will alert the operator.
- If running Java applications, note that Java Virtual Machines (JVMs) are loaded in MVS storage, requiring a considerable amount of storage there (100 MB+).

Using LPA / ELPA



LPA usage provides :
Economy of real storage
Protection
Reduced path length

Reentrant Code:	LPA usage specified via:
Most CICS nucleus modules	SIT : LPA=YES
Many non-nucleus CICS modules and application programs	SIT : LPA=YES and RDO definition DEFINE PROGRAM(pgm) USELPACPY(YES)

Figure 11-4. Using LPA / ELPA CI207.0

Notes:

The benefits of placing reentrant or shareable code in the Link Pack Area are:

- Reduced real storage requirements, if there are multiple CICS regions.
- Faster CICS startup, since the modules would already be loaded.
- Storage protection, since modules in the LPA use a different storage protect key from those in user address spaces.

Any link-edited reentrant program is a candidate for LPA/ELPA residency.

- CICS nucleus programs have to be copied from SDFHAUTH and SDFHLOAD, respectively, to an LPA-library.
- For CICS non-nucleus modules and user programs, LPA usage is activated via SIT and RDO parameters.
- The *CICS Transaction Server for z/OS Version 3.1 Installation Guide, GC34-6426,* contains very detailed documentation about what programs place in the LPA/ELPA and how to do it.

Region Residency

CEDA/CEDB
DEF PROG( )
or
DEF MAP( )

	Storage Reserved	Program Loaded	Storage Freed
RES(NO) USAGE(Normal)			Not recently used DSA space needed
RES(NO) USAGE(Transient)	First use		No users
RES(YES)			Only if CEMT newcopy or phasein
RELOAD(YES)	Every use		User responsibility

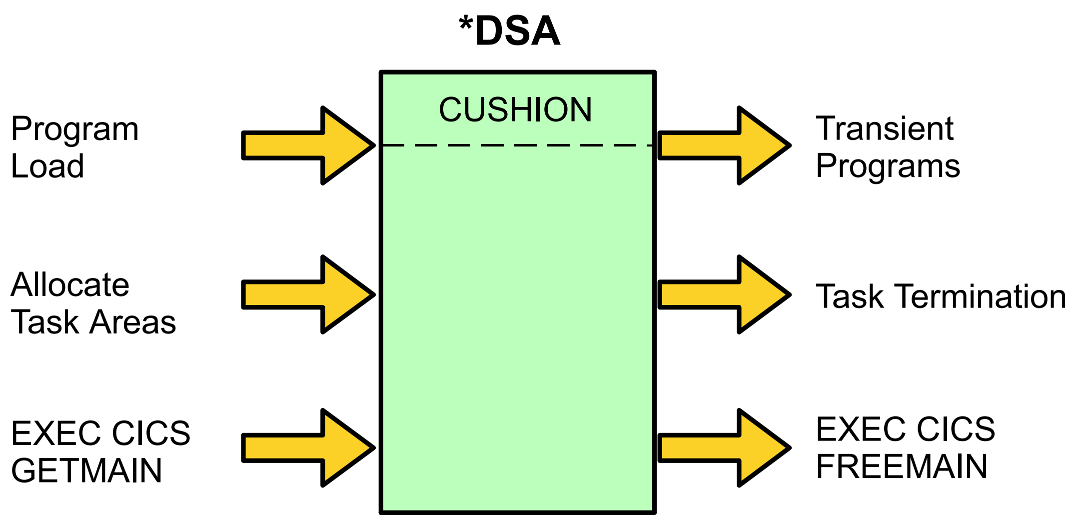
Figure 11-5. Region Residency CI207.0

Notes:

This chart summarizes residency options in the CICS region, which may be determined with program and map set resource definitions. The goal of residency is to reduce program fetch I/O.

- The default, `RES(NO)` and `USAGE(NORMAL)`, allows CICS total freedom to load the module modules as needed and free them on a least-recently-used basis.
- The `TRANSIENT` option might be appropriate for rarely-used programs or map sets.
- CICS only deletes programs defined with `RES(YES)` if you load a new copy.
- Even if no action is taken, CICS TS keeps high-use programs resident, to avoid reloading them. Use the default definitions, unless there are compelling reasons to do otherwise.
- The definitions supplied for CICS programs that handle emergencies — abends, errors, and backout — already specify `RES(YES)`. Do not change these.
- CICS can take advantage of LLA, since it uses the MVS loader.
- `RELOAD(YES)` has to be coded for modules that require a fresh copy for each execution, and it is the responsibility of the application to issue an EXEC CICS FREEMAIN for the program after use.

Short On Storage - SOS



SOS:
No new user tasks attached
Raise priority of "older" tasks
No logons accepted

DFHSM0131 Short On Storage below the 16MB line
DFHSM0133 Short On Storage above the 16MB line

Figure 11-6. Short On Storage - SOS CI207.0

Notes:

- DSAs are filling up during CICS operation, mainly with loaded programs.
- CICS automatically balances the single DSA's portion of the DSALIM/EDSALIM size and calculates fitting cushion sizes.
- Whenever a request for storage cannot be satisfied at all, or only at the cost of cushion storage, a program compression is performed: releasing the storage that is occupied by not-in-use programs.
- If, though compressed, a certain DSA does not contain enough free storage to satisfy on outstanding request for storage, the **Short-on-Storage** (SOS) condition is raised for this DSA.
- If SOS situations occur, you will have to identify the reason for this. The basic approach is to determine whether the problem is that just too many tasks are active, or if a single task or a few tasks are consuming too much storage.
  So, the first action normally is to decrease the MXT value, and if the problem persists, try to identify the tasks that are active at SOS time.

11.2 Storage Protection

Preventing Storage Violations

Read-Only Areas	Separating Systems and User Areas	Separating Task Areas from each other
Read-Only Key-0 Storage	Subsystem Storage Protection	Subspace Group Facility
RDSA,ERDSA	CDSA,ECDSA for CICS (system) UDAS,EUDSA for application (user)	Allocating (E)UDSA for task lifetime storage in different subspaces

SIT:
`RENTPGM=PROTECT`
	SIT:
	`STGPROT=YES`
	Transact.-Def:
	`TASKDATAKEY=CICS|USER`
		SIT:
		`TRANISO=YES`
		Transact.-Def:
		`ISOLATE=YES`

Figure 11-7. Preventing Storage Violations CI207.0

Notes:

A badly-written application may cause a storage violation. Depending on the data overlaid, various symptoms can result: task abend, invalid data handled, or CICS abend.

To avoid these negative issues, CICS TS for z/OS will use the storage protection facilities available with the z/OS architecture.

Each environment where CICS TS for z/OS can be installed and run supports all storage protection facilities, and is made up of three mechanisms:

1. **Using Key 0 (read-only) storage for reentrant programs:**
   Some kind of “LPA inside the CICS region”.
2. **Subsystem storage protection**:
   Application (user) code may only READ but *not* UPDATE System (CICS) areas.
3. **Allocating task lifetime storage from different subspaces per task:**
   Thus isolating transactions from each other.

Read-Only DSAs

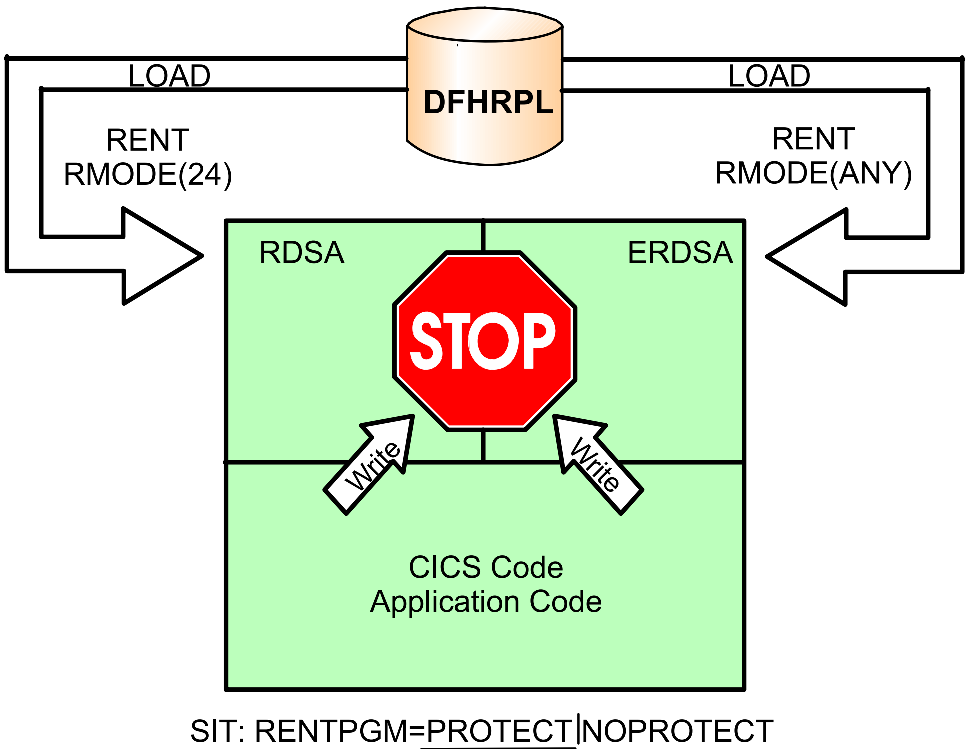


Figure 11-8. Read-Only DSAs CI207.0

Notes:

- Activated by the SIT parameter **RENTPGM=PROTECT**.
- Programs, CICS and application, that are link-edited as re-entrant, are loaded into read-only areas RDSA (24-bit programs), or ERDSA (31-bit programs), respectively.
- Some kind of “LPA inside the CICS region”.
- Write attempts to PROTECTed RDA/ERDA area result in a program check, issued as an ASRA abend for application tasks.
- Can be activated independently from the other storage protection functions.
- Storage is allocated to RDSA and ERDSA within the limits specified by SIT parameters DSALIM and EDSALIM in extents of 1 MB.

Storage Protection

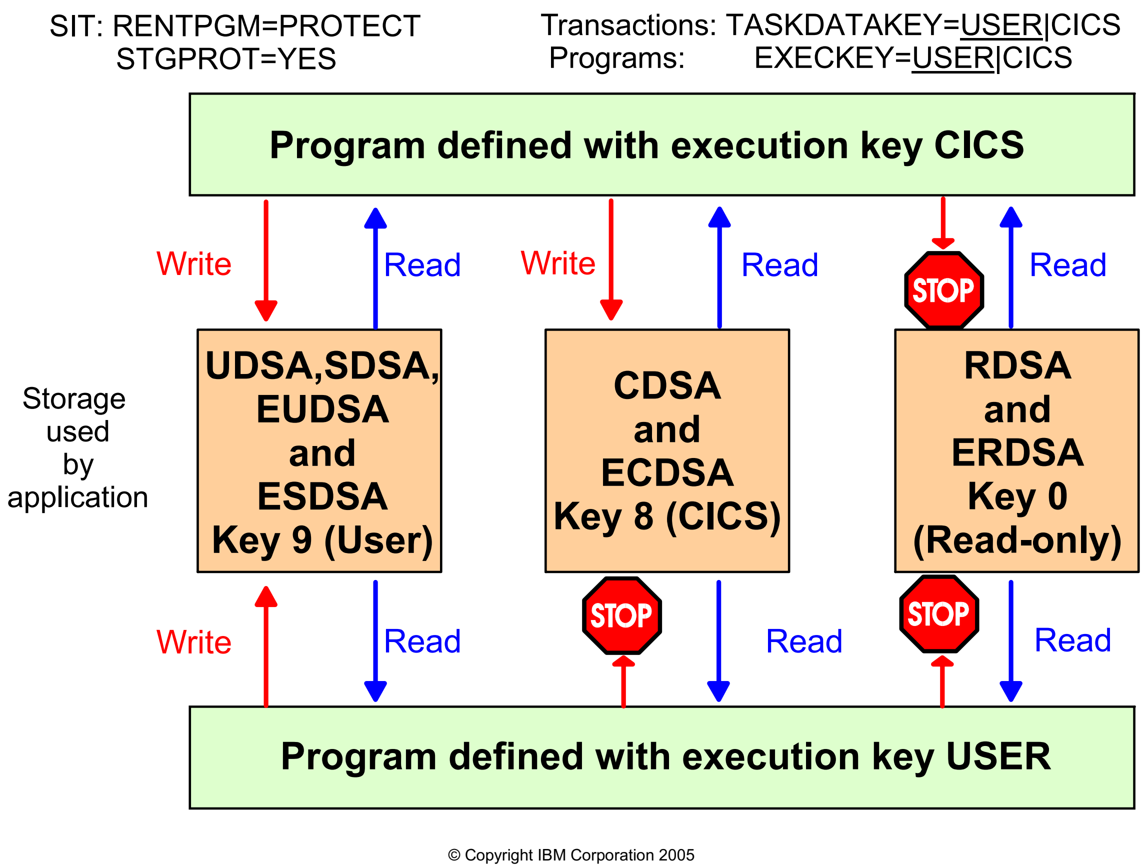


Figure 11-9. Storage Protection CI207.0

Notes:

Two kinds of execution key are available: CICS and USER, with different write authority.

- Programs executing in CICS key can:
  a. Read and update data in the User, Shared, and the CICS DSA.
  b. Read data in the Read-only DSA.
- Programs executing in USER key can:
  a. Read and update data in the User and Shared DSA.
  b. Read data in the both the CICS and the Read-only DSA.
- Write attempts by USER programs against CICS areas result in program check/ASRA and message DFHSR0001.

Two kinds of storage key: User for UDAS/EUDSA and SDSA/ESDSA; CICS for CDSA/ECDSA.

- The shared DSAs, SDSA and ESDSA, are used to load:
  - Non-reentrant link-edited programs with EXECKEY(USER)

  - Shared storage with TASKDATAKEY(USER)
- UDSA and EUDSA contain task-lifetime storage for tasks with TASKDATAKEY(USER).
- CDSA and ECDSA are used to load:
  - Non-reentrant link-edited programs with EXECKEY(CICS)
  - CICS control blocks
  - Task-lifetime storage for tasks with TASKDATAKEY(CICS).
- RDSA/ERDSA usage remains unchanged.

So, the aim is to run as many application programs as possible with EXECKEY(USER).

Transaction Isolation Principle

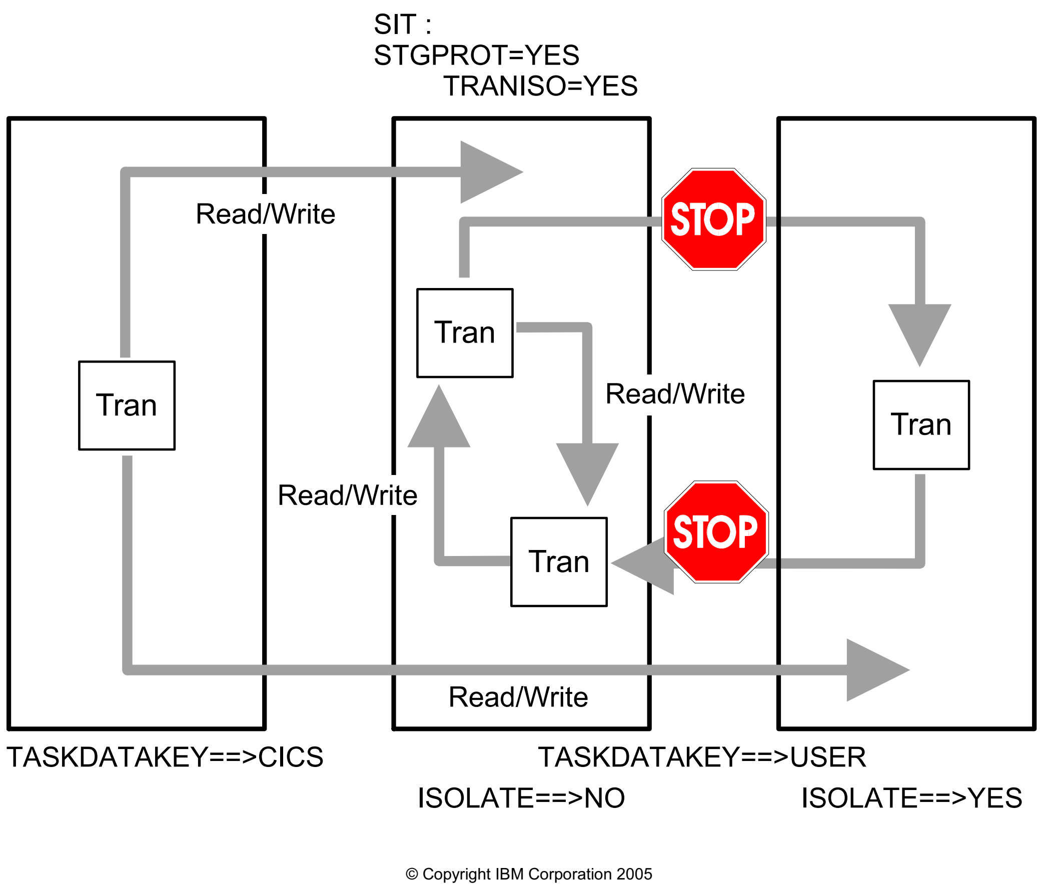


Figure 11-10. Transaction Isolation Principle CI207.0

Notes

- Storage protection must be activated.
- Only transactions executing in USER key can be isolated from each other:
  - USER-key tasks run in separate subspaces, if transaction isolation is requested. Task-lifetime storage is protected between tasks and against not-isolated USER-key tasks.
  - USER-key tasks with ISOLATE=NO specified run in a single common subspace. These may share task-lifetime storage among each other, but are protected against isolated tasks running in unique subspaces.
- For `TRANISO=YES`, the UDSA will be allocated in 1 MB increments, aligned on a 1 MB boundary, as required by the MVS subspace group facility.
  The other DSAs below the 16 MB line will still be allocated in 256 KB increments.
  - *Note: with transaction isolation active, (E)UDSA increments are required on a per-task basis; thus the need for DSA storage may increase considerably.*

DSA Definitions

Program Definition	Transaction Definition	Link Edit attributes		Storage areas for ...	
EXECKEY	TASDATAKEY	AMODE	RENT	Program Load	Default GETMAIN
		ANY	Yes	ERDSA	
		24	Yes	RDSA	
CICS	CICS	ANY	No	ECDSA	ECDSA
CICS	USER	ANY	No	ECDSA	EUDSA
USER	CICS	(either)	(either)	invalid combination	
USER	USER	ANY	No	ESDSA	EUDSA
CICS	CICS	24	No	CDSA	CDSA
CICS	USER	24	No	CDSA	UDSA
USER	USER	24	No	SDSA	UDSA

Figure 11-11. DSA Definitions CI207.0

Notes:

- If you specify TASKDATAKEY(CICS) for a transaction, an attempt to run a program in user key (EXECKEY(USER)) under this transaction will abend the task with abend AEZD.
- The default for both values is USER.
- The default can be overridden by coding the type of DSA on a GETMAIN request.

Dynamic DSA Management

SIT: EDSALIM=30M

```
CEMT INQ DSAS
  .
  .
  .
EDsalimit ( 0031457280 )
ECdsasize (0004194304)
ERdsasize (0019922944)
ESdsasize (0000000000)
EUdsasize (0001048576)
```

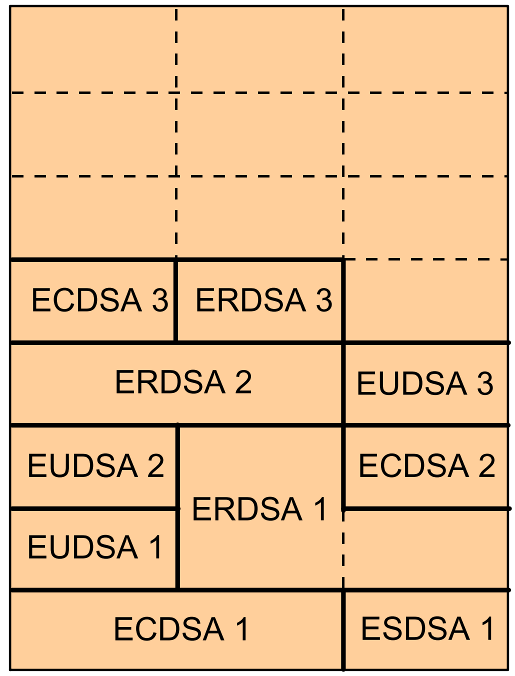


Figure 11-12. Dynamic DSA Management CI207.0

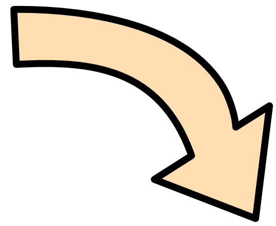
Notes:

- From the storage specified by the DSALIM and EDSALIM SIT parameters, DSA “extents” will be allocated automatically by CICS as needed.
  - Multiple extents for a certain DSA need not be contiguous.
  - EDSAs are allocated in 1 MB increments, aligned on a 1 MB boundary.
  - DSAs are allocated in 256 KB increments for `TRANISO=NO`.
  - For `TRANISO=YES`, the UDSA will be allocated in 1 MB increments. The other DSAs below the 16 MB line will still be allocated in 256 KB increments.
  - If there is no requirement for SDSA, or ESDSA, or both, these areas are not allocated, and size (0) is displayed when inquired.
  - DSALIM and EDSALIM may be changed online within the region boundary.
- Fix sizes for one or more DSAs may be specified by the use of optional SIT override parameters `(E)*DSASZE=`. This will reduce storage fragmentation and avoid all the storage being allocated to one DSA.

Further Storage Protectable Areas

SIT: CWAKEY={USER|CICS}
TCTUAKEY={USER|CICS}

CMDPROT={YES|NO}



For example: EXEC CICS READFILE (...) INTO (data-area)
EXEC CICS READ TSQUEUE (...) INTO (data-area)

Figure 11-13. Further Storage Protectable Areas CI207.0

Notes:

- The default storage location for both the CWA and TCTUA is user storage.

  If protection of these areas is required, code “CICS” to protect them from USER-key programs.

- CMDPROT=YES

  CICS validates that the task has write access to the storage area referenced in the CICS program.

  A validation failure results in a AEYD task abend.

Storage Management (Summary)

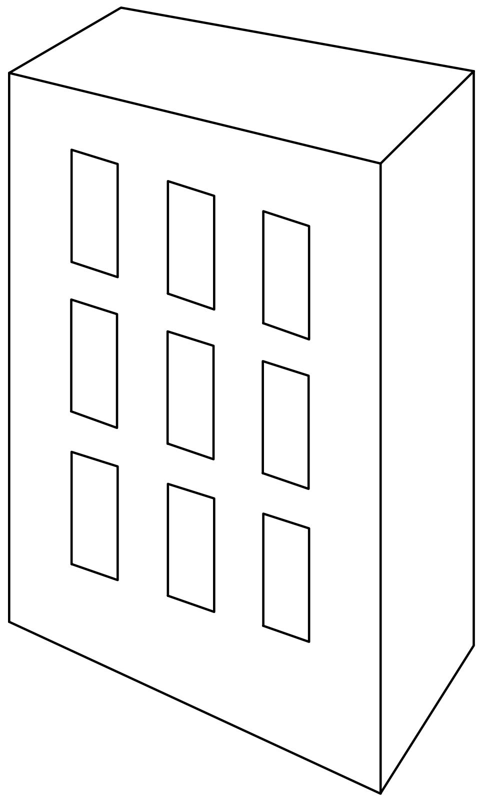


Figure 11-14. Storage Management (Summary) CI207.0

Notes:

- The amount of virtual storage you need depends on a variety of factors, for example:
  - Workload or transaction rate and application programs' size
  - Number of CICS resources defined
  - Use of the LPA for reentrant modules
- For multiple CICS regions executing within one MVS system, central storage can be conserved using the LPA/ELPA for reentrant CICS modules and programs, and user programs.
- Defining some programs and map sets as resident in the CICS region may reduce program fetch I/O.
- The storage protection function introduced with CICS 3.3 separated system areas from user areas, thus strengthening system availability.
- Using the MVS (V5) subspace group facility, task-lifetime storage can be isolated between transactions.

- Program and transaction definitions determine:
  - What DSA is used for programs
  - The default task storage allocated
- Only the maximum storage area that is obtained as DSA by CICS is specified. CICS manages storage allocation for the different DSAs automatically.

Unit Summary

Having completed this unit, you should be able to:

- List the main areas of the CICS address space
- State the main variables affecting CICS region size
- State the benefits of using the Link Pack Area, and define CICS and user programs as resident in the LPA
- Define region residency options for application programs and map sets
- Describe the features that protect CICS DSA storage against storage violations available in CICS TS for z/OS
- Activate storage protection and transaction isolation by use of SIT and resource definition parameters

Figure 11-15. Unit Summary CI207.0

Notes:

Unit 12. Startup and Recovery/Restart

What This Unit Is About

This unit teaches how the way in which CICS is shut down determines how you can restart. It will introduce the handling and definition of recoverable resources, and the use and definition of z/OS log streams for logging and journaling.

What You Should Be Able to Do

After completing this unit, you should be able to:

- Describe the ways of terminating CICS and the corresponding methods of restarting
- Describe the use of the PLT and XLT in shutdown and restart
- State the SIT parameters required for proper recovery and restart of the system
- Describe the purpose of the Activity Keypoint and Warm Keypoint records
- Define the z/OS Coupling Facility structures required for the CICS TS system log and for forward recovery logs
- Identify the options used to define recoverable files and queues

How You Will Check Your Progress

- Machine lab exercise 8.
  A set of log streams is defined for each of the team CICS regions, and the exercise is to provide CICS definitions in order to use these log streams, and to redefine application resources as recoverable. As in the previous lab exercises, all definitions created by students may be verified immediately.

References

CICS Transaction Server for z/OS Version 3.1 Installation Guide, GC34-6426

CICS System Definition Guide, SC34-6428

CICS Resource Definition Guide, SC34-6430

CICS Recovery and Restart Guide, SC34-6246

Unit Objectives

After completing this unit, you should be able to:

- Describe the ways of terminating CICS and the corresponding methods of restarting
- Describe the use of the PLT and XLT in shutdown and restart
- State the SIT parameters required for proper recovery and restart of the system
- Describe the purpose of the Activity Keypoint and Warm Keypoint records
- Define the z/OS Coupling Facility structures required for the CICS TS system log and for forward recovery logs
- Identify the options used to define recoverable files and queues

Figure 12-1. Unit Objectives CI207.0

Notes:

12.1 Initial Start and Execution

Startup and Recovery/Restart Management

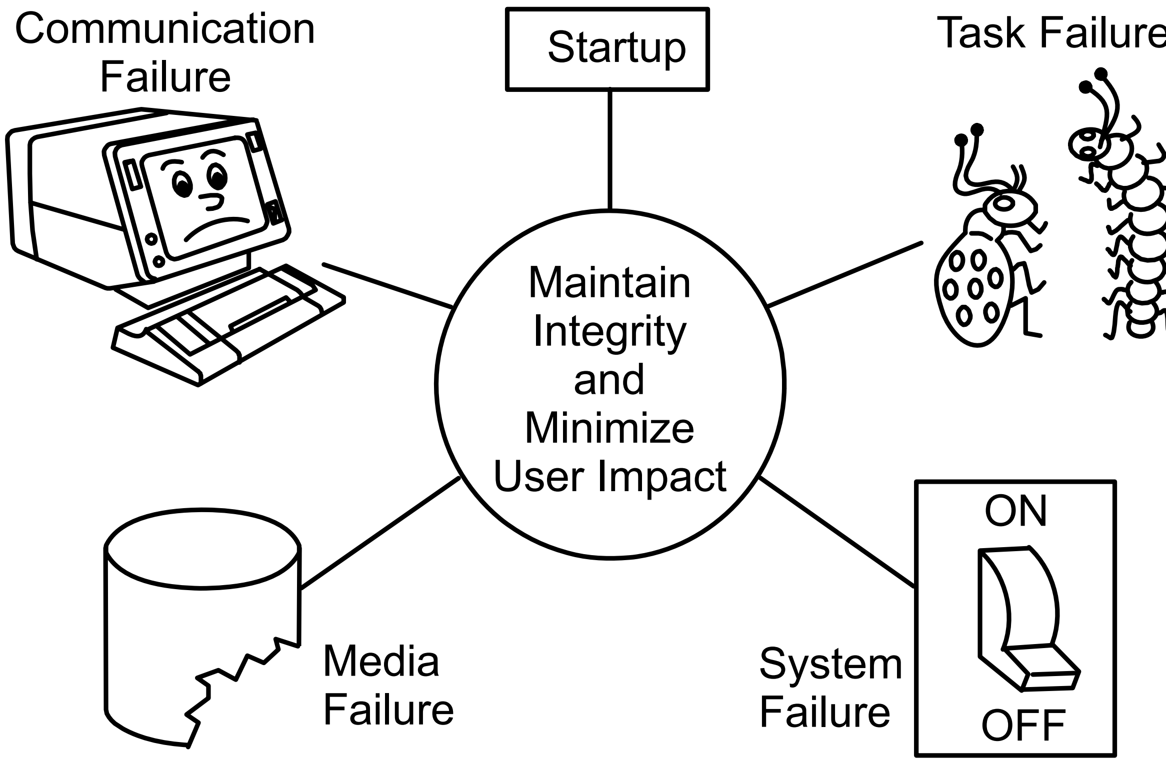


Figure 12-2. Startup and Recovery/Restart Management CI207.0

Notes:

The CICS TS Recovery Manager Domain and Restart Management provide protection from these types of failure in order to protect user data.

CICS TS Initial and Cold Start

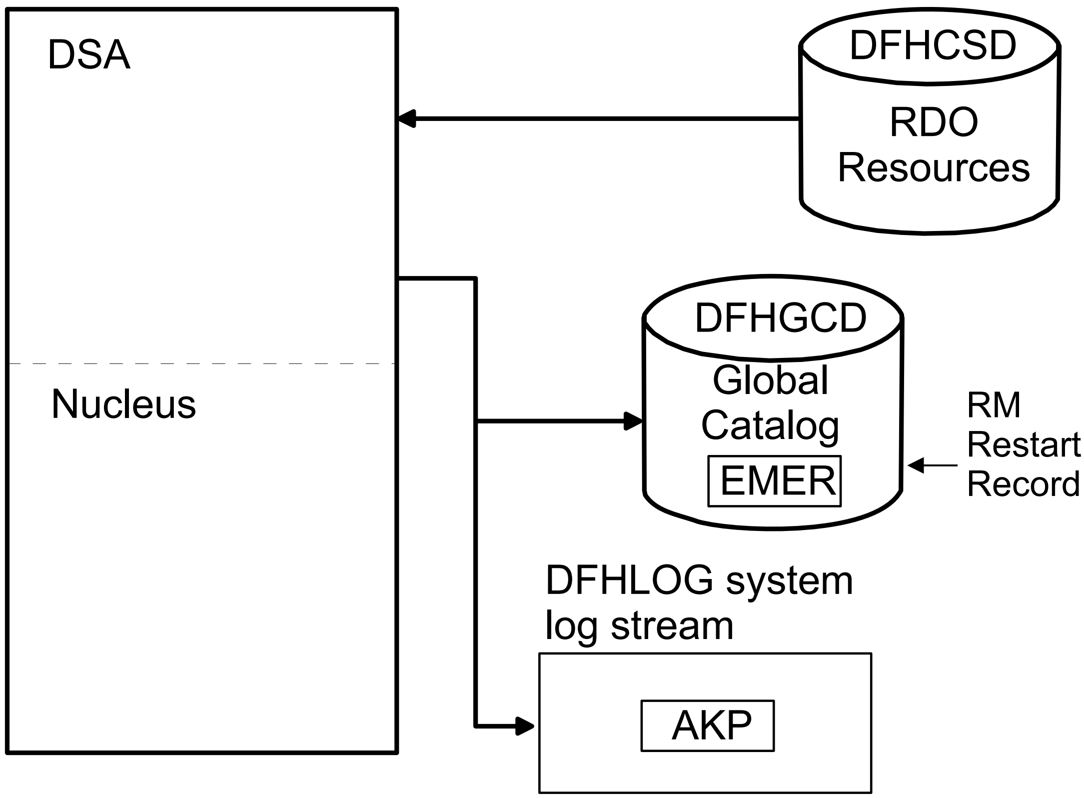


Figure 12-3. CICS TS Initial and Cold Start CI207.0

Notes:

- During execution, restart information is written to the Catalog Data sets and the system log, which is a log stream structure of the coupling facility or on a DASD device.
- To start CICS with a completely “clean state”, without any reference to the status of resources during the last execution, **INITIAL** start the system.
  - Resource definitions are loaded from the DFHCSD data set, as coded by the GRPLIST parameter in the SIT.
  - A new system log is allocated, while the old one is ignored.
- A **COLD** start loads resource definitions in the same way, and ignores system log information about the state of **locally-owned** recoverable resources, but preserves all information needed to resynchronize with remote partners, such as remote CICS systems and external resource managers.
- When initialization is complete, CICS writes an “emergency restart needed” indicator into the **global catalog data set (DFHGCD).**

- You have to define and initialize the DFHGCD data set, together with the DFHLCD Local Catalog Data set, as described in the “Defining and Using Catalog Data Sets” chapter of the *CICS System Definition Guide, SC34-6428*, and include a DD statement for it in the CICS job stream.

  The post-installation job, DFHDEFDS, generated by the DFHISTAR facility, contains the JCL and control statements to perform this on a per-region basis as a part of the initial region setup.

- CICS also writes an **Activity Keypoint (AKP)** record on the system log at startup. More on this later.

Program List Table - PLTPI

```
                 PLTPI

DFHPLT   TYPE=INITIAL,SUFFIX=PI
DFHPLT   TYPE=ENTRY,PROGRAM=USERPROG
DFHPLT   TYPE=ENTRY,PROGRAM=DFHDELIM
DFHPLT   TYPE=ENTRY,PROGRAM=DFHD2CM0
DFHPLT   TYPE=ENTRY,PROGRAM=DFHDBCON
DFHPLT   TYPE=FINAL
```


Figure 12-4. Program List Table - PLTPI CI207.0

Notes:

- CICS loads the **Post-Initialization Program List Table (PLTPI)** during startup.
- The programs are executed in the listed sequence and must not use terminal commands.
- The `DFHDELIM` entry divides the list into two parts:
  - The sole function of programs located prior to `DFHDELIM`, like USERPROG, is to enable CICS global user exits.
  - Phase 1 — PLT Programs must be linked `AMODE(ANY)`.
  - Programs in the second part may perform housekeeping, such as starting the DB2 and DL/I-DBCTL database interfaces or attachments in previous releases of CICS. This is the function of `DSN2COM0` and `DFHDBCON`.
- PLT programs may be secured by means of SIT parameters `PLTPISEC` and `PLTPIUSR`.
- For details of the PLT and other macro-assembled tables used for recovery and restart, see the *CICS Resource Definition Guide, SC34-6430*.

Logging for Recovery

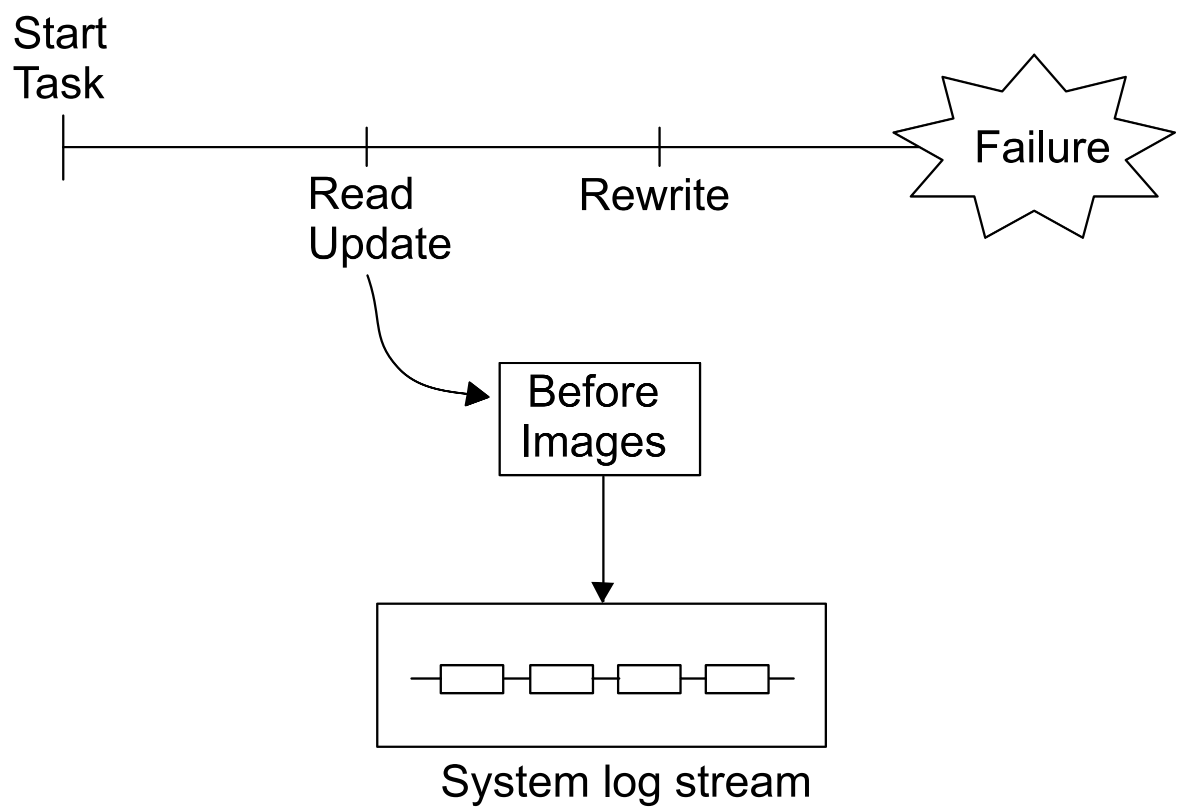


Figure 12-5. Logging for Recovery CI207.0

Notes:

- Recall that CICS is logging changes to recoverable resources to the system log to support backout, using the services of the z/OS logger.
- The **Before Images** of data records that were changed by a unit of work are required for backout if this Unit of Work (UoW) fails.
- The CICS system log is implemented by a separate log stream for each CICS region.
  - All Before Images of a region's unit of work processing are written to this System Log log stream, as requested by the appropriate CICS resource managers.
  - Log stream blocks are written in the time order in which the appropriate resources are accessed, but UOW-related log records are chained together.
  - On task termination, the **Recovery Manager Domain** (RM) coordinates syncpoint processing between the resource managers involved in the UoW, and requests the removal of all the chained log stream block of the UoW, when all changes are committed.

Backout of Abending Transactions

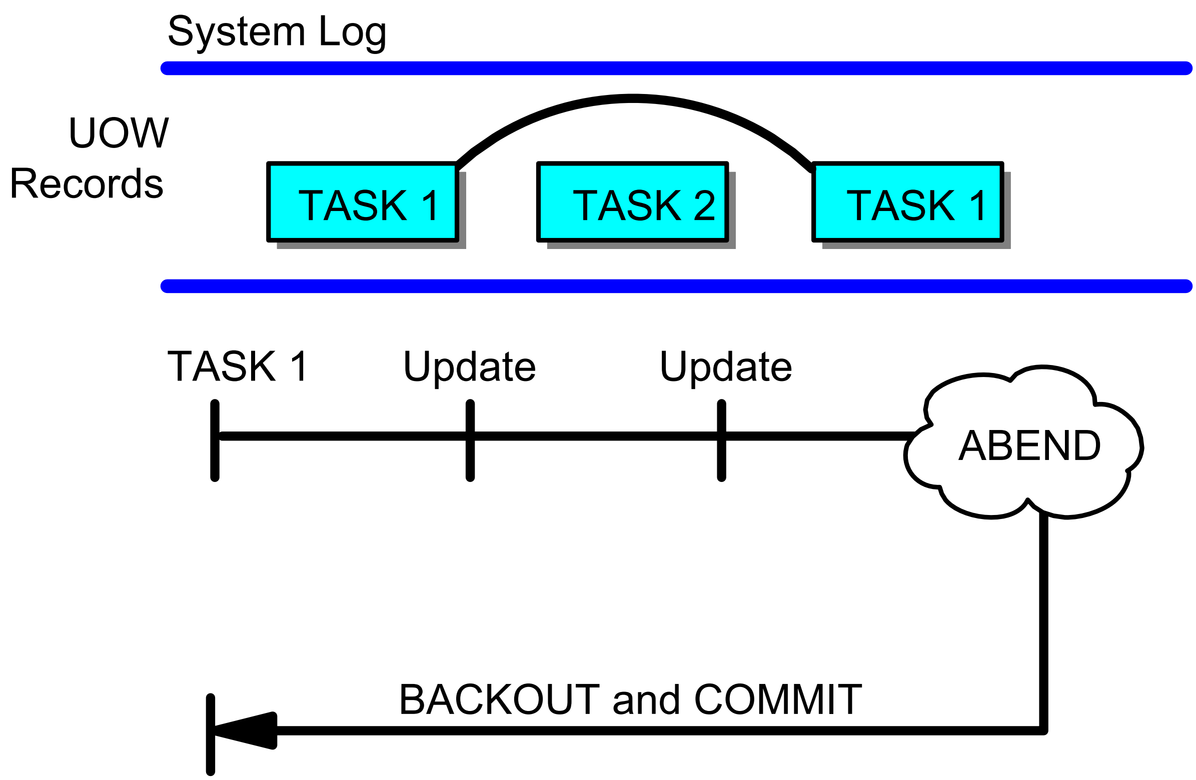


Figure 12-6. Backout of Abending Transactions CI207.0

Notes:

- Records can be retrieved from the system log stream directly as well as sequentially.
- Thus, if a task (transaction) fails or the `EXEC CICS ROLLBACK` command is issued by the application, log records for an active UOW can be identified directly.
- The CICS TS Logger Domain retrieves the appropriate log records, and performs the backout to the recoverable resources affected, thus completing the unit of work. Sometimes, the term *Dynamic Backout* is still used for this operation.
- When backout is completed, the appropriate UOW's records are no longer needed and are marked for deletion from the system log stream, just as the log records written on behalf of units of work (transactions) that completed regularly.

Note: Non-UOW-related logging (forward recovery logging and automatic journaling, and so on) is not performed by the Recovery Manager Domain.

12.2 Termination of CICS TS for z/OS

Normal Termination

```
CEMT PERFORM SHUTDOWN
       XLT(01) PLT(02) SDTRAN(CESD)
```

- **Quiesces running tasks**
- **User can be involved**
- **Writes statistics data to SMF**
- **Closes files**
- **Saves system status**
- **Sets warm start indicator**

Figure 12-7. Normal Termination CI207.0

Notes:

- This is the recommended way to shut down CICS, because it occurs in a controlled sequence.
- All tasks must complete before shutdown can proceed.
- A normal shutdown is the **only** time CICS sets Recovery Manager's Control Record in the global catalog data set to indicate that a warm start is possible.
- Normal system shutdown is a three-stage process.
- The parameters of the shutdown command, which all are optional, are covered by the following visuals.

Stage 1 for Normal Termination

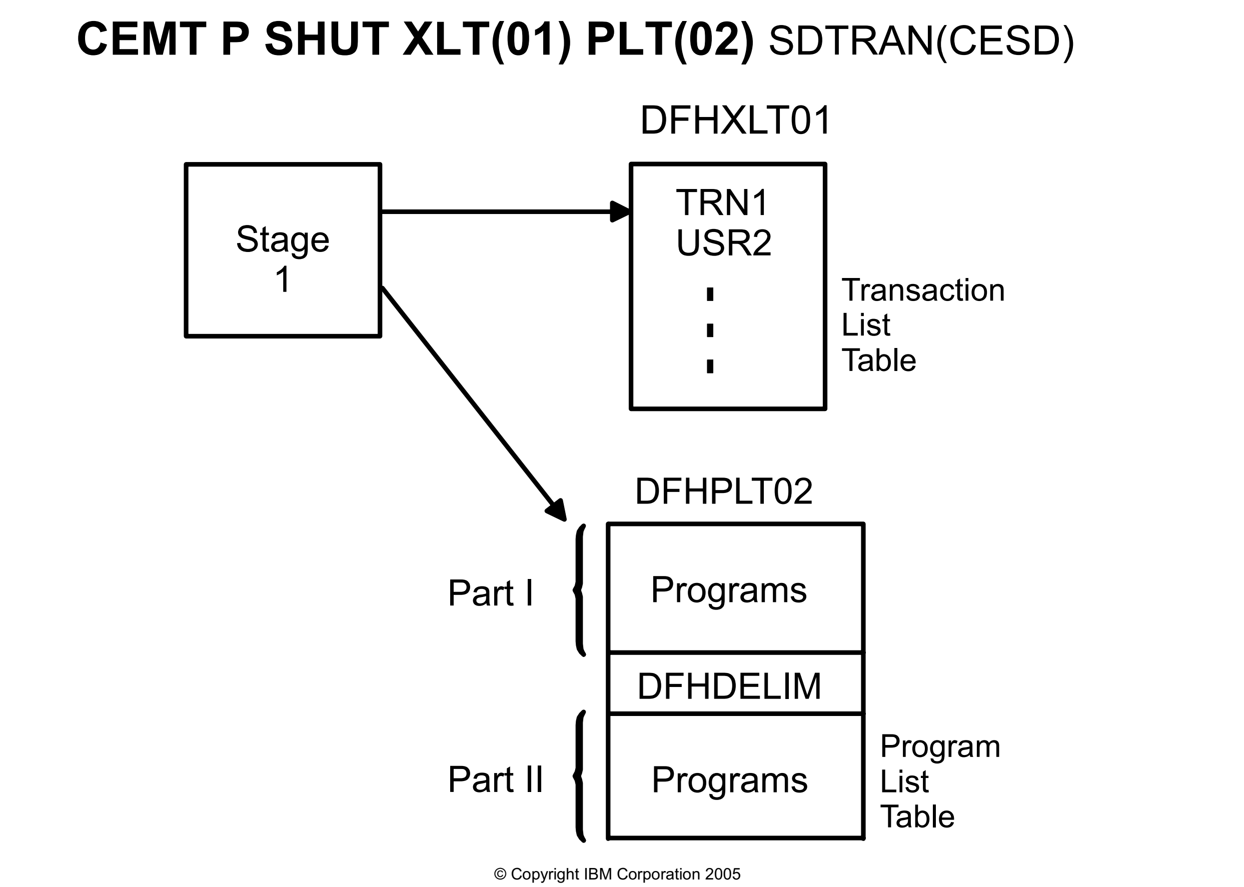


Figure 12-8. Stage 1 for Normal Termination CI207.0

Notes:

- The Transaction List Table (XLT) is a list of transactions that operators can initiate during stage 1 quiesce.
- The shutdown PLT (PLTSD) lists programs you want CICS to execute in sequence during stages 1 and 2 of system quiesce. Programs in part 1 may use terminal commands.
  - This could be a good way to send a message informing users that the system is coming down. See the *CICS Customization Guide, SC34-6429*, chapter “Writing Initialization and Shutdown Programs” for more details.
- Stage 1 is complete when all tasks have terminated and all programs in Part 1 of the PLTSD (up to DFHDELIM) have executed.

Shutdown XLT Example

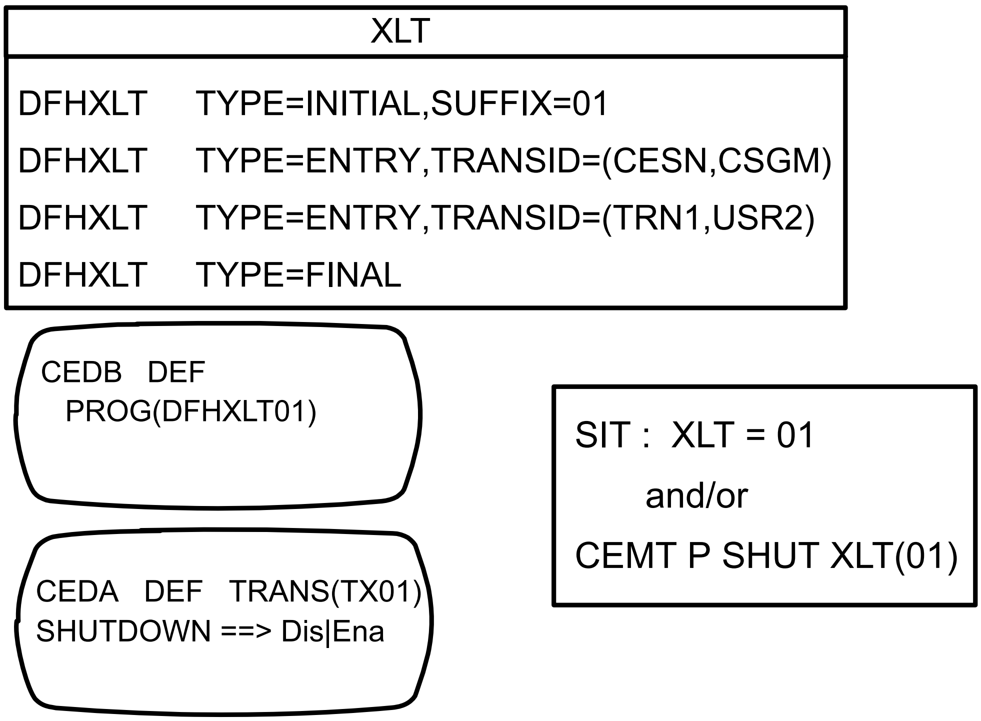


Figure 12-9. Shutdown XLT Example CI207.0

Notes:

- The XLT includes user and CICS transactions you want to allow during stage 1 of shutdown, for example, the IDs of pseudoconversational transactions needed to complete work in process.
- CICS will allow entry of transactions such as CEMT and CESF, regardless of your XLT specification.
- The DFHXLT... loadable module must be made known to CICS via a static program definition or by the program autoinstall function.
- *Supplementary* to XLT, allowance to be initiated during stage 1 quiesce may be specified through the `SHutdown` parameter of the transaction definition.

Why is it important to have access to CEMT at this point?

Shutdown PLT

PLTSD

```
DFHPLT    TYPE=INITIAL,SUFFIX=02
DFHPLT    TYPE=ENTRY,PROGRAM=USERPROG
DFHPLT    TYPE=ENTRY,PROGRAM=DFHDELIM
DFHPLT    TYPE=FINAL
```

```
CEDB   DEF
  PROG(DFHPLT02)
```

```
SIT :  PLTSD = 02
      and/or
CEMT P SHUT PLT(02)
```


Figure 12-10. Shutdown PLT CI207.0

Notes:

- CICS executes shutdown PLT programs in the order you define them.
- The PLT has to be assembled and linked. Since CICS loads it, it must also be defined to CICS as a program resource, or it has to be program autoinstalled.
- The PLTSD suffix may be specified in either or both of two places:
  a. The SIT
  b. The CEMT shutdown command, which also overrides any SIT specification.

Shutdown Assist Transaction

CEMT P SHUT XLT(01) PLT(02) **SDTRAN(CESD)**

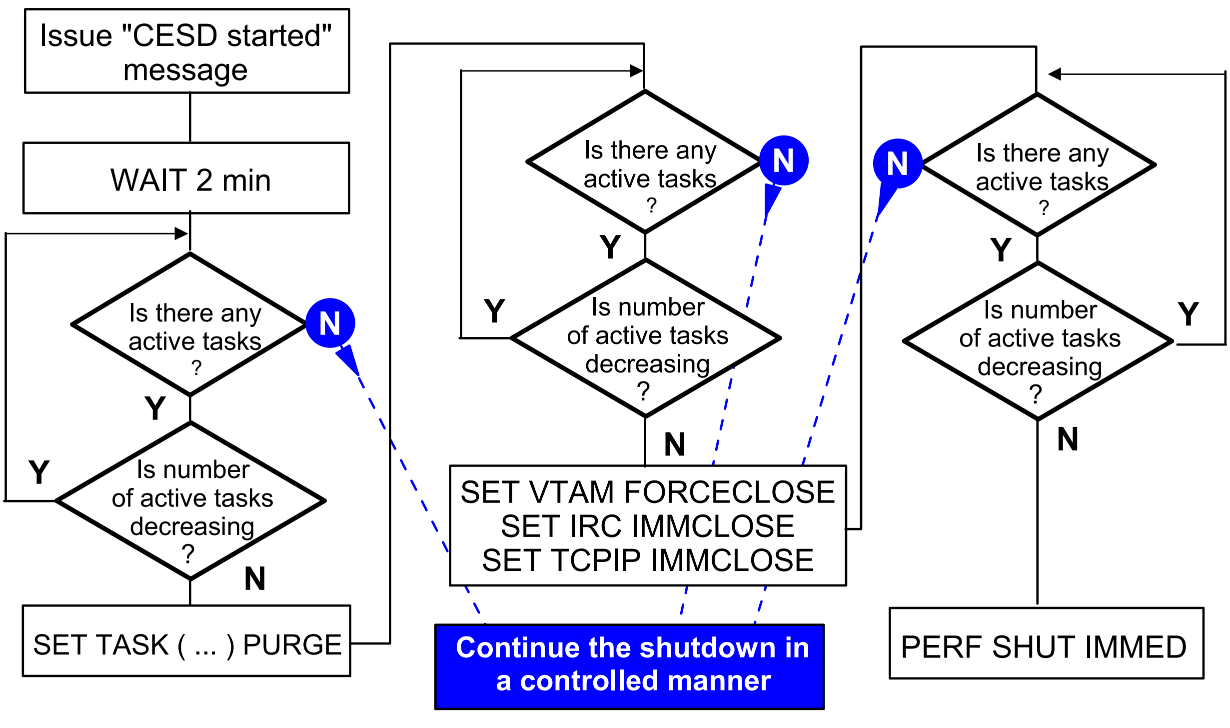


Figure 12-11. Shutdown Assist Transaction CI207.0

Notes:

- A Shutdown Assist Transaction may be specified, which is meant to purge long-running tasks that won't quiesce in a reasonable time. This invokes backout processing for these UOWs, but lets CICS shut down cleanly.
- The Shutdown Assist Transaction executes under the userid of the CEMT transaction that invoked the shutdown.
- The CICS-provided default Shutdown Assist Transaction, **CESD**, is to ensure that as many tasks as possible commit or backout cleanly in a reasonable time.
  - CESD lists its activity, issuing messages to the console.
  - If, after all purging actions, there are still tasks active in the system, DFHCESD will shut down CICS through an `E.C.PERFORM SHUTDOWN IMMEDIATE` command.
  - Source code is in `SDFHSAMP(DFHCESD)`.
- `NOSDTRAN` specified on the P SHUT command or `SDTRAN=NO` specified in the SIT lets CICS wait indefinitely for all active tasks to terminate.
- The same is true if the program named by the shutdown transaction cannot be loaded.

Stage 2 for Normal Termination

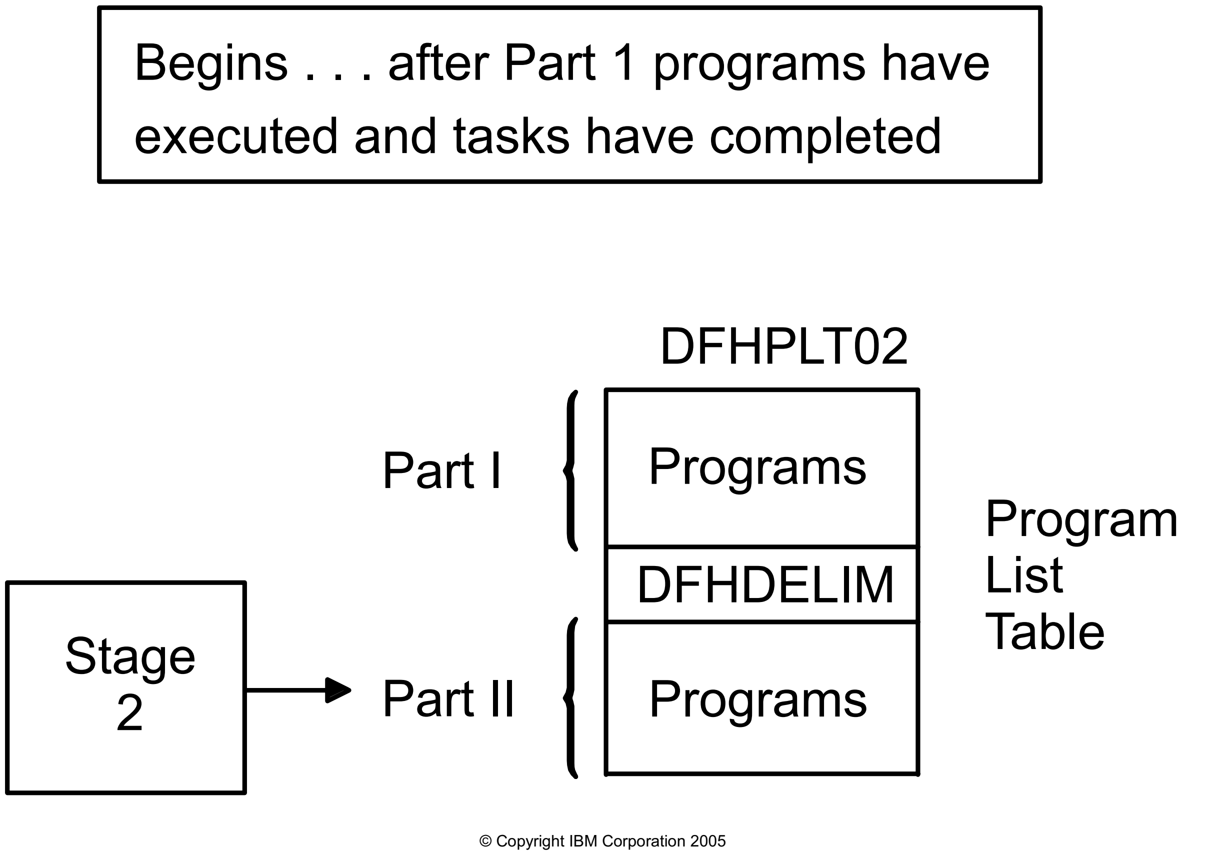


Figure 12-12. Stage 2 for Normal Termination CI207.0

Notes:

- Stage 2 simply executes programs in the shutdown PLT after DFHDELIM.
- Programs in the second part of the PLTSD cannot use terminal services.

Stage 3 for Normal Termination

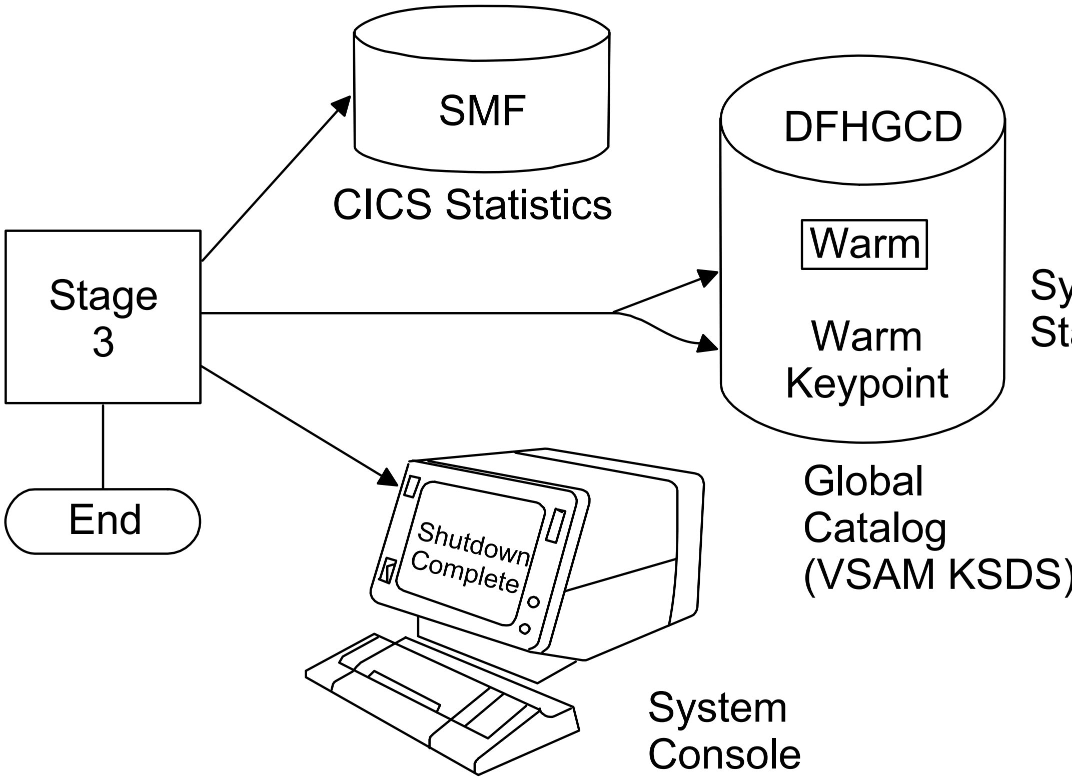


Figure 12-13. Stage 3 for Normal Termination CI207.0

Notes:

CICS performs these functions in the final stage of termination:

- Writes end-of-day statistics to SMF
- Closes files
- Writes a warm keypoint record to the Global Catalog to provide information for a warm start
- Turns on the “warm-start-possible” indicator to indicate that a normal termination occurred

For more information on the resource and control information in the warm keypoint, see the “CICS Startup” chapter of the *CICS Recovery and Restart Guide, SC34-6246*.

Immediate Termination

CEMT PERFORM SHUTDOWN IMMEDIATE

- **Maybe incomplete tasks**
- **No PLT/XLT invoked**
- **Shutdown assist allowed**
- **Writes statistics data to SMF**
- **Files not closed**
- **No keypoint or warm start indicator written**

Figure 12-14. Immediate Termination CI207.0

Notes:

- Immediate termination is some kind of terminating in a “quick and dirty” way, but note that CICS still knows what it does, so it's a form of controlled shutdown.
- There is no staged shutdown, and the shutdown PLT and XLT are not used.
- The Shutdown Assist Transaction, however, may be specified, as for normal shutdown.

Since the restart indicator is not changed, what type of restart will be required?

Assist Transaction on Immediate Shutdown

CEMT P SHUT - EXEC CICS PERF SHUTDOWN
IMMEDIATE SDTRAN(CESD)

- **2-minute wait period is skipped**
- **Otherwise, same processing as with normal shutdown**

Figure 12-15. Assist Transaction on Immediate Shutdown CI207.0

Notes:

- On immediate shutdown, without using a shutdown assistance transaction, CICS abends running tasks and does **not** perform backout until emergency restart. This may cause an unacceptable number of resources to remain in a locked state, either in CICS or in connected systems.
- Use of CESD (or a modification of it) will largely solve this problem.
- As on a normal shutdown, if the program named with the shutdown assistance transaction cannot be loaded, CICS waits indefinitely for all user tasks to complete, which may cause shutdown to hang.
- During normal shutdown processing, CESD invokes an Immediate Shutdown if one or more tasks cannot be PURGEd over a longer time. This does NOT cause CESD to run again, as CICS only permits one use of the shutdown assist transaction mechanism.

MVS-Requested Termination

Invokes System Recovery Program

SIT: SRT=1$ (recommended)
or SRT={YES|NO|xx}

Allows MVS system dump if DUMP=YES in SIT

Similar to immediate termination

Figure 12-16. MVS-Requested Termination CI207.0

Notes:

- An MVS abend will not always bring CICS down, because CICS traps many MVS abend requests associated with user applications.
  - If the abend code is included in the System Recovery Table (SRT), CICS usually abends the offending task with an ASRB dump, but keeps the region up.
  - If the code is not included, the region continues to abend.
- Unless you have coded your own SRT, specify SRT=1$, the sample provided with CICS. It supports recovery from several MVS abend codes, which are listed in the *CICS Resource Definition Guide, SC34-6430*.
- An MVS abend of the CICS region has the same restart implications as an immediate shutdown.

There will be no shutdown statistics, however.

Uncontrolled Termination

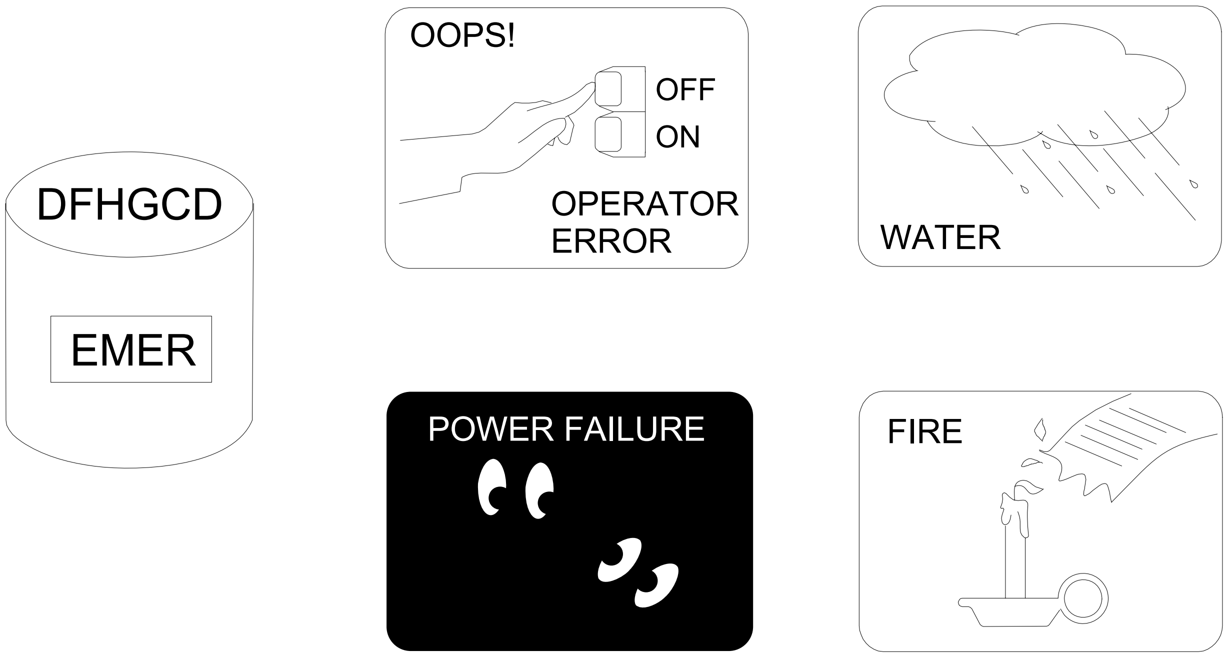


Figure 12-17. Uncontrolled Termination CI207.0

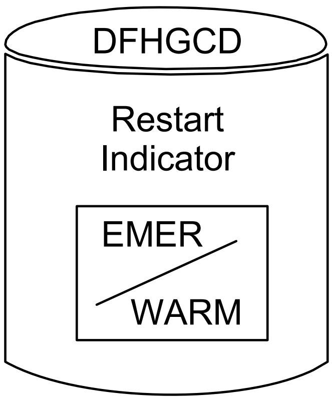
Notes:

- Since CICS never gets control in this type of situation, there is no chance to prepare for restart, and the emergency-restart-needed indicator is not reset.

12.3 Restart Considerations

CICS Startup Options

- Automatic restart processing
- SIT parameter: START=AUTO|COLD|INITIAL



DFHSIT	Restart Indicator	Start Type
START=INITIAL	N/A	INITIAL
START=COLD	N/A	COLD
START=AUTO	EMER	EMER
START=AUTO	WARM	WARM
START=AUTO	ABSENT	INITIAL

Figure 12-18. CICS Startup Options CI207.0

Notes:

- START=AUTO provides either warm or emergency restart, depending on the setting of the restart indicator.
- If the catalog data sets have just been initialized, there is no indicator on the DFHGCD, so CICS will automatically perform an INITIAL start.
- If the Global Catalog fails to open, a GO or CANCEL message will be issued.
  - CANCEL terminates CICS.
  - GO causes an INITIAL start of CICS.

DFHRMUTL Utility

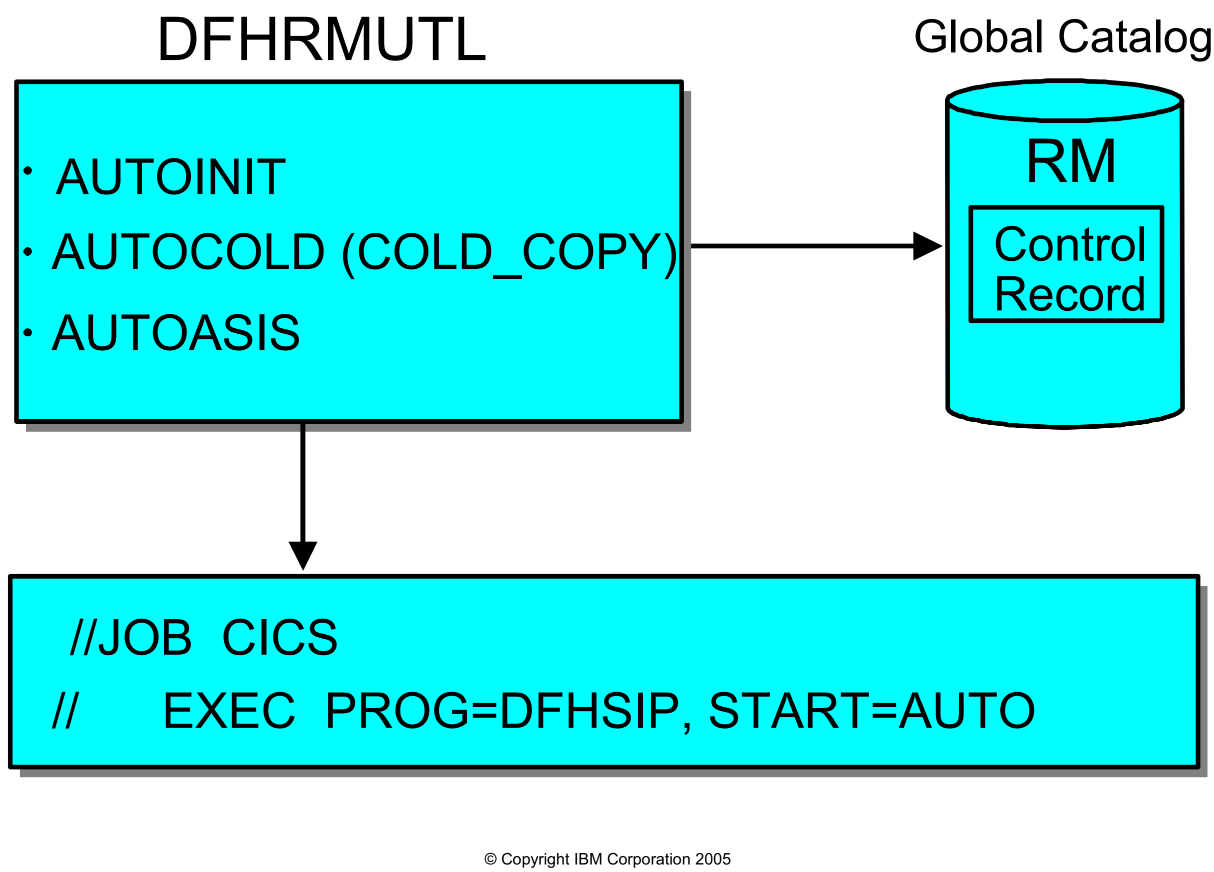


Figure 12-19. DFHRMUTL Utility CI207.0

Notes:

- DFHRMUTL processes the global catalog data set. It can insert or modify the recovery manager autostart override record. Optionally, it can extract a subset of the catalog records to build a reduced new catalog for a cold start.
- Options to be coded are:

AUTOINIT	Perform an INITIAL start.
AUTOCOLD	Perform a COLD start.
AUTOCOLD COLDCOPY	Create in a NEWGCD a copy of only those records from DFHGCD that CICS needs in order to perform a cold start. This will improve cold start times.
AUTOASIS	No change to the RM's control record.

- This utility is meant to be used within automated environments, in order to determine the way CICS is restarting by means of a preconfigured jobs, instead of requiring operator actions. The DFHRMUTL utility returns information about the type of previous CICS shutdown, which is of use in determining whether a cold restart is possible or not. For further details, see the *CICS Operations and Utilities Guide, SC34-6431*.

Warm Start (Restart) Options

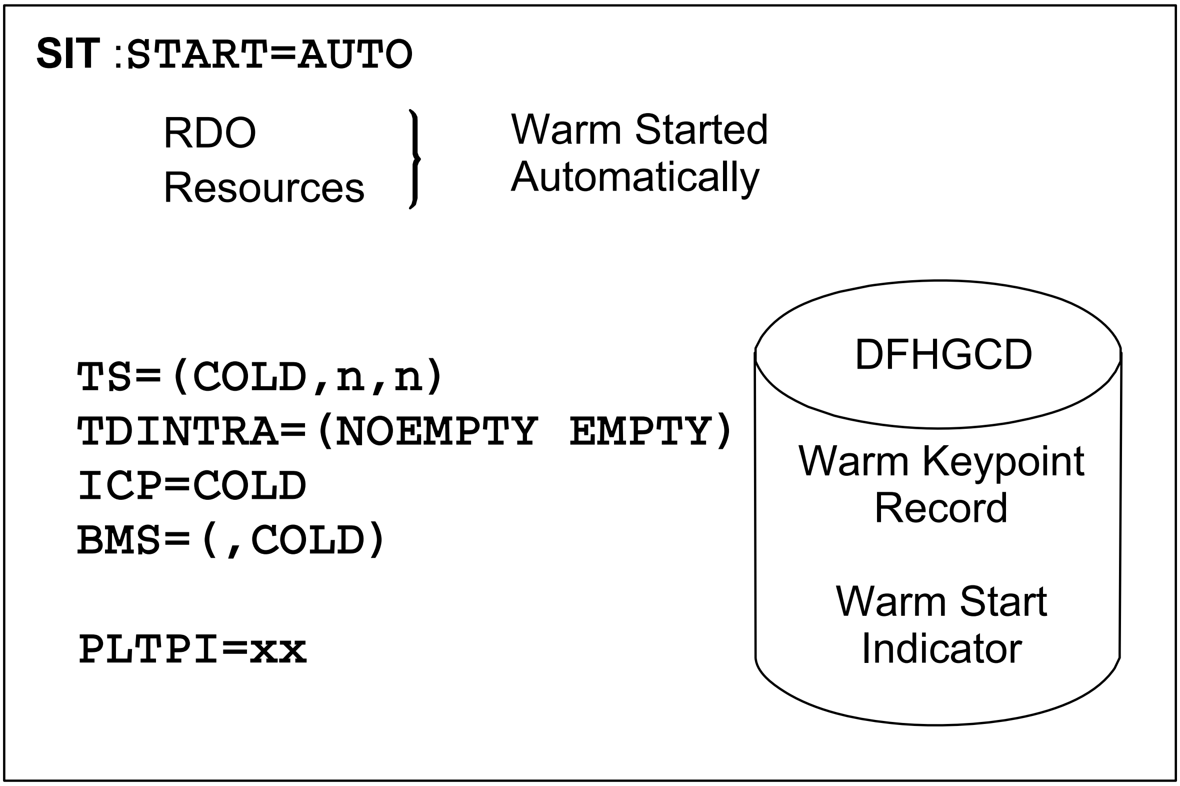


Figure 12-20. Warm Start (Restart) Options CI207.0

Notes:

- The contents of CICS system data sets, such as DFHINTRA, DFHTEMP, DFHDMPA/B, and DFHA/BUXT, are normally preserved across a warm start.
- Some individual facilities can be COLD started selectively.
  - `TS=(COLD,...)` tells CICS to release all queues on the Temporary Storage data set. `TDINTRA=EMPTY` is the equivalent option for the Transient Data function. `ICP=COLD` causes CICS to discard Interval Control Elements (ICE) that represent outstanding `EXEC CICS START` requests from the previous run, and `BMS=(,COLD)` unconditionally deletes delayed messages from temporary storage.
- Warm start normally preserves SIT values that have been changed during CICS execution, such as MXT and your statistics recording interval. If you wish to restart CICS with the values you specified in the SIT, instead of the way they were at shutdown, use the `NEWSIT=YES` override.
- All resource definitions are retrieved from GCD, with the status they had at the previous shutdown time.

SIT Parameter Overrides

Parm String of EXEC Statement

```
//CICS EXEC PGM=DFHSIP,
            PARM='SIT=A1,PLT=P1,DSALIM=5M,....SI,CN'
```

SYSIN-Dataset or Instream Data

```
//SYSIN DD *
      TS=(COLD,5,5)
      GMTEXT='HELLO
     .END
/*
```

Console Input by Operator

after prompt message

```
DFHPA1104 applid SPECIFY ALTERNATIVE SIT PARAMETERS, IF ANY,
                 AND THEN TYPE '.END'.
```

SIT: **PARMERR=INTERACT|ABEND|IGNORE**

Figure 12-21. SIT Parameter Overrides CI207.0

Notes:

- Parameters specified for override will take effect at each kind of restart.
- Certain parameters may only be specified at certain points (for example, security parameters cannot be overridden via console) — for detailed information, see the *CICS System Definition Guide, SC34-6428*.
- SIT parameters:

  ```
  FCT=
  GRPLIST=
  CSD*=
  ```

  can only be changed via cold start.
  Overrides will not be honored if specified on a restart other than cold start.
- The CICS parameter domain checks the SIT override parameters. The **PARMERR** option identifies what happens when a misspelled or out-of-range parameter or value is detected.

System Recovery Preparation

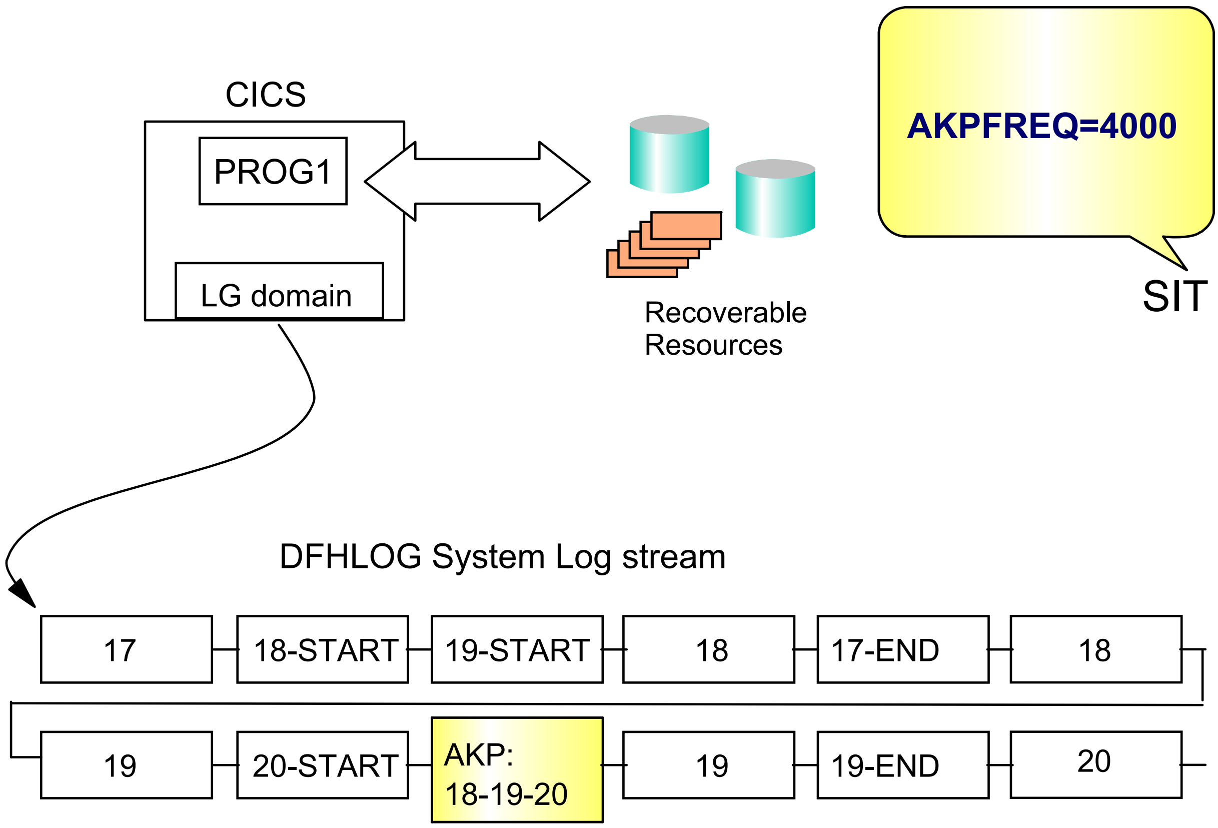


Figure 12-22. System Recovery Preparation CI207.0

Notes:

- CICS logs back out information for recoverable resources to the system log stream. In addition to the “before image” of the resource, CICS includes the task number (actually, the unit-of-work ID), and a flag if this was the starting or ending log record for the task.
- Periodically, CICS writes an Activity Keypoint (AKP) record containing, among other things, the system control information about what tasks are currently active.
- Use SIT parameter **`AKPFREQ`** to determine the frequency of writing these records. The value specifies how many blocks are written to the system log stream buffer between activity keypoints. For heavily loaded regions, a time interval in the range of a few seconds is desirable between AKPs.

Log Tail Management - DFHSHUNT

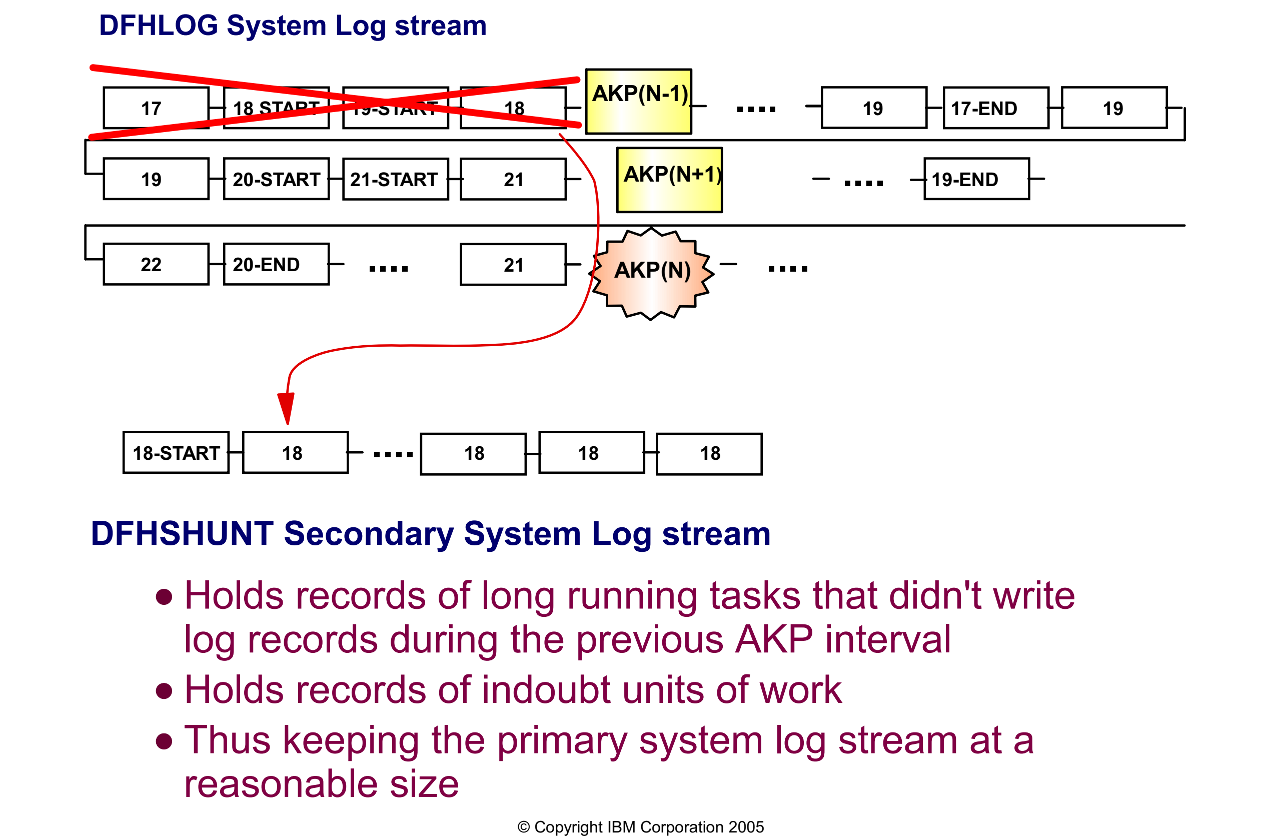


Figure 12-23. Log Tail Management - DFHSHUNT CI207.0

Notes:

- In addition to writing the IDs of active units of work to the system log, the system log is cleaned up at activity keypoint time, a process which is called **log tail management**:
  - The intention is to delete all records that are older than two AKP periods from the DFHLOG log stream. This can be done by a simple call to the logger.
  - There may, however, be tasks that are active for a longer time than two AKP cycles. Records of these tasks may still be required for backout in the future, and must not be lost. These records are moved from the DFHLOG log stream to a secondary log stream named **DFHSHUNT**, if that task didn’t have log activity for two checkpoint cycles, and logging for those tasks is directly written to the DFHSHUNT log stream thereafter.
- Without writing activity keypoints (`AKPFREQ=0`), or if there are very long-running tasks with frequent log activity, system log data is not cleaned up, which may cause emergency restart to be greatly delayed, because CICS will have to read back the entire system log stream, in order to identify the units of work that were **in-flight** at the time of a system abend.

SHUNTED Unit of Work State

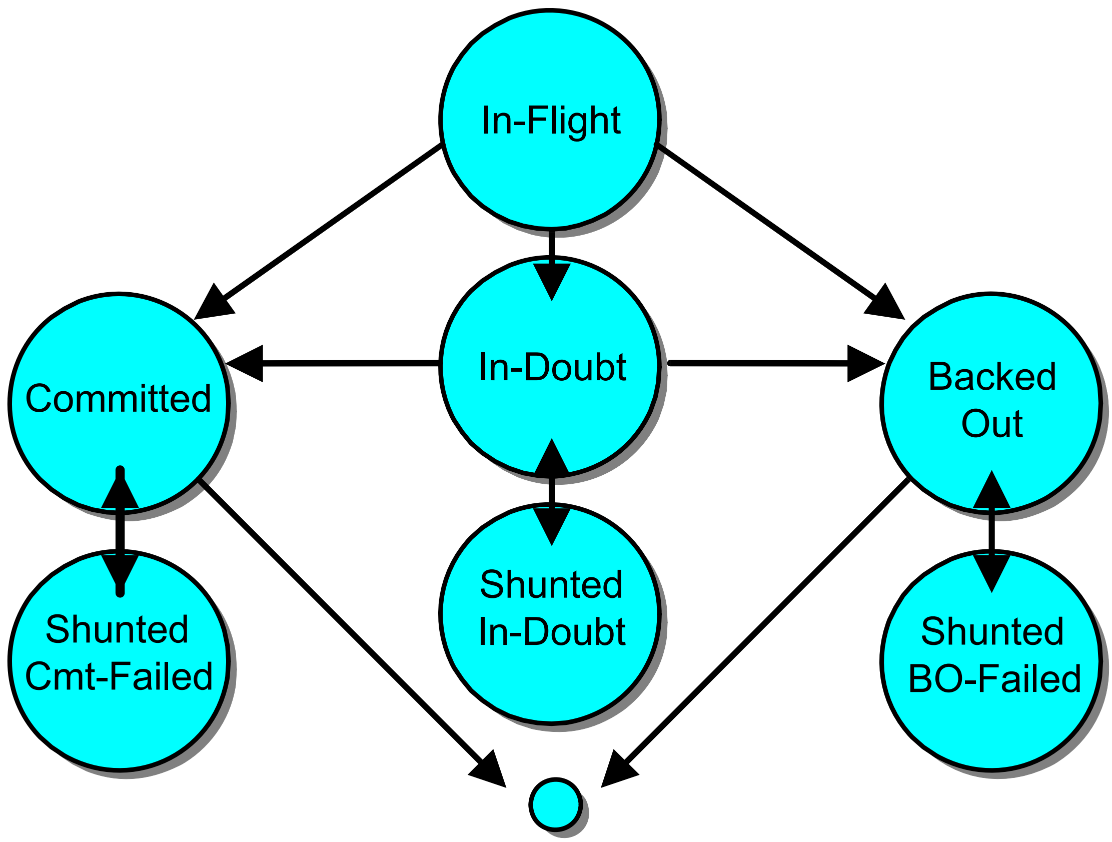


Figure 12-24. SHUNTED Unit of Work State CI207.0

Notes:

- Distributed Unit of Work (UoW) may enter a **SHUNTED** state if one of the following happens:
  - If an in-doubt UOW loses connection to its coordinator, or the coordinator fails.
  - If commit processing fails.
  - If backout processing fails.
- Note that UoWs that use function shipping, VSAM RLS, and/or TS Data Sharing are distributed UoWs.
- In each of these cases, the RM domain moves the log records written for the appropriate UoW from the DFHLOG stream to the DFHSHUNT stream.
- Active locks owned by these UoWs are converted to **RETAINED**.
- Remember that the log records of long-running tasks are moved to the DFHSHUNT log stream as part of the activity keypoint processing. These UoWs, however, do not enter the `SHUNTED` state.

Emergency Restart

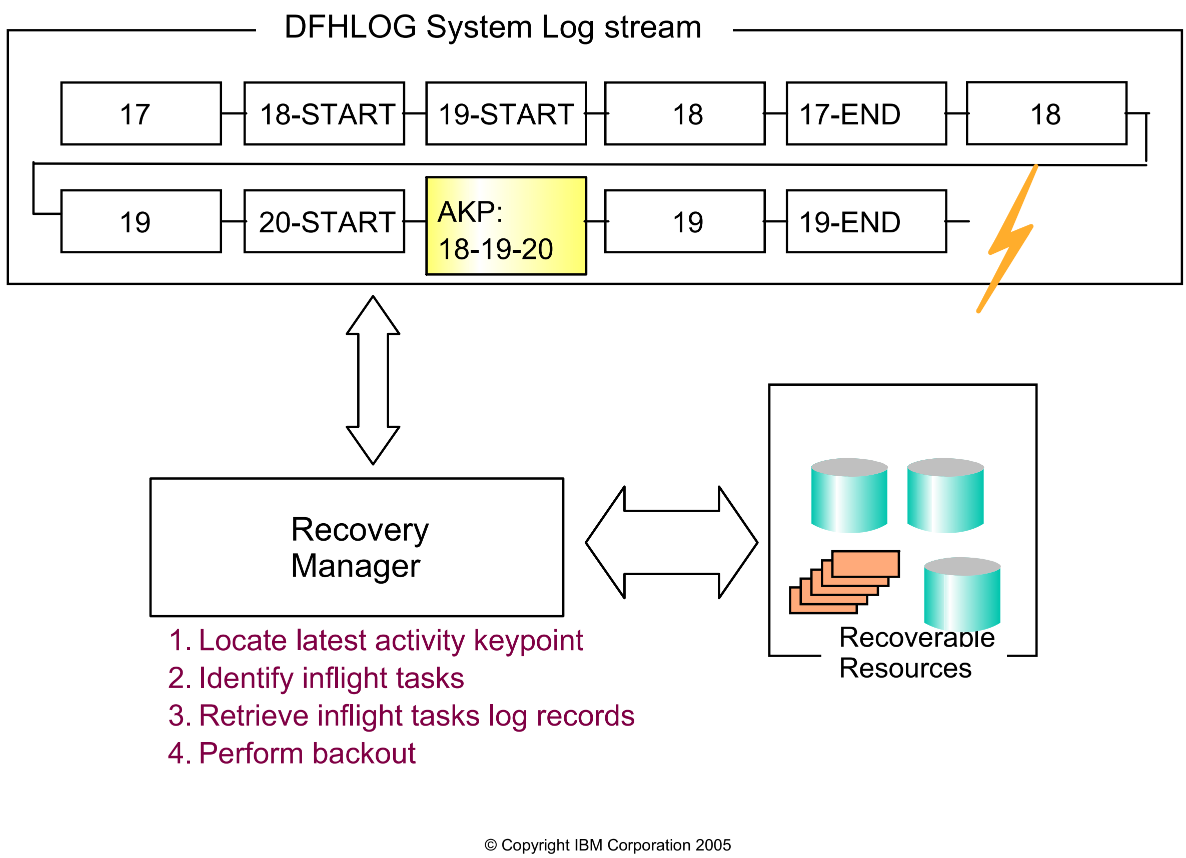


Figure 12-25. Emergency Restart CI207.0

Notes:

- If **`START=AUTO`** is specified, and the emergency-restart-needed indicator is set in the global catalog, CICS does an emergency restart.
- The CICS Recovery Manager does the following:
  - Searches for the last activity keypoint on the system log
  - From the information found there, it identifies the tasks (UoWs) that were active at the time when CICS abended
  - Then, it retrieves these UOWs' log records from DFHLOG and (if applicable) DFHSHUNT
  - Eventually, it uses this information to restore all data and the region itself to the committed state at the time of the abnormal termination.
- User exits may be supplied in order to control the backout process. For details, refer to "User Exits for Transaction Backout" in the *CICS Recovery and Restart Guide, SC34-6246*, and the *CICS Customization Guide, SC34-6429*.

12.4 Handling CICS Log Streams

System Logger Overview

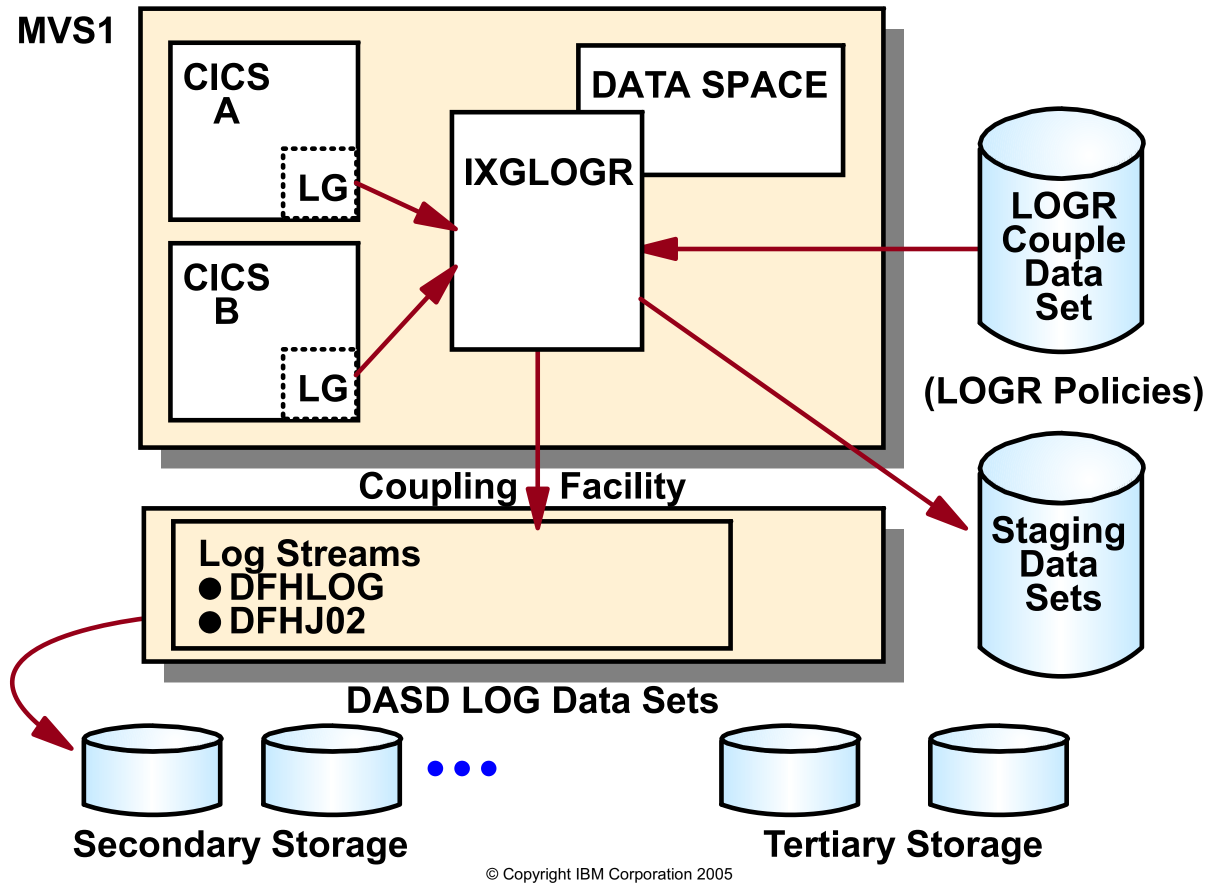


Figure 12-26. System Logger Overview CI207.0

Notes:

- The MVS system logger is the manager of a collection of **log streams**, and provides a programming interface to access records on those log streams.
- A log stream is a sequence of data blocks, identified by its log stream identifier, the log stream name.
- Log streams may be written to list structures of the coupling facility, or to DASD data sets as primary storage (DASD-only logging requires OS/390 V2R4 and CICS TS 1.2 or above).
- Basically, each log stream is accessible from any address space in the sysplex. However, a DASD-only log stream may be opened by one MVS system only at a time, which is not true for CF log streams.
- All log streams are logically independent objects, but for log streams written to coupling facility storage, multiple log streams may be part of one **log structure** in the CF.
- For DASD-only logging, each log stream is represented by a separate data set.

Log Stream Storage for CF Logging

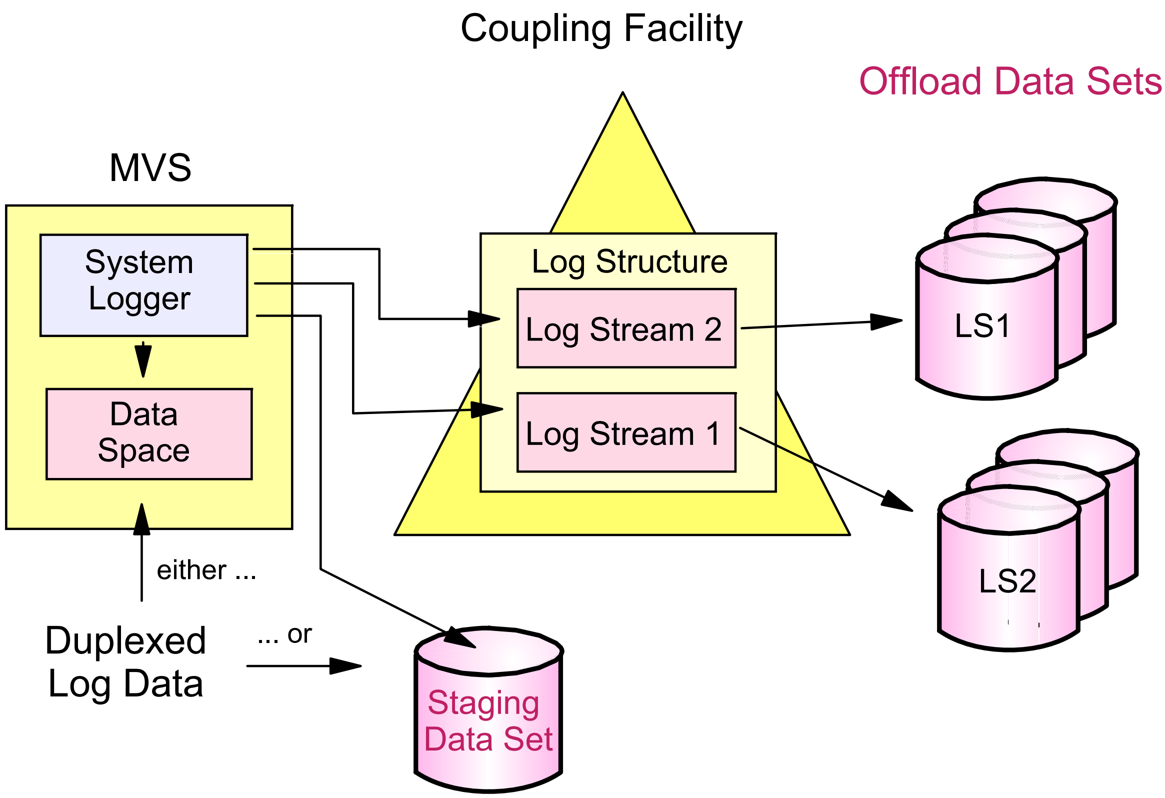


Figure 12-27. Log Stream Storage for CF Logging CI207.0

Notes:

The following is true if a coupling facility is available for logging.

- A log stream's **primary storage** is a structure within the coupling facility, which holds the most recent records written to the log stream. For data security reasons, log data written to the coupling facility is also copied to either a data space or a **staging data set**.
- When the primary storage structure for a log stream becomes full, the older records automatically spill into **secondary storage**, which consists of data sets managed by SMS. This process is known as *DASD offloading*.
- **Tertiary storage** is optional and is used as specified to HSM. It can be either DASD data sets or tape volumes, to which older log stream data sets may be migrated.

Log Stream Storage for DASD-Only Logging

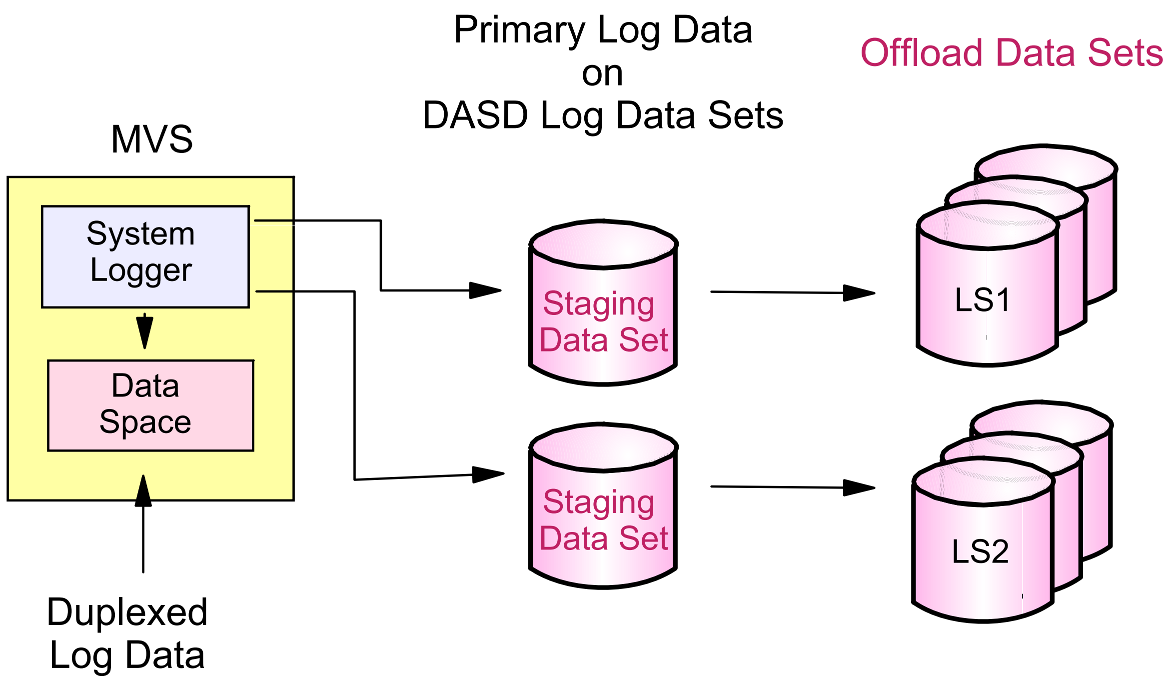


Figure 12-28. Log Stream Storage for DASD-Only Logging CI207.0

Notes:

Log blocks that are written to a DASD-only log stream:

- Are first written by the system logger to the local storage buffers of the system that reside in data spaces owned by the z/OS system logger.
- Then, they are automatically duplexed to a DASD staging data set associated with the log stream.
- When the staging data set runs out of space, it is offloaded in the same way as coupling-facility-defined log stream, as is the processing of the secondary and tertiary storage data.
- If you have a coupling facility available and are running V2R4 of OS/390, you may decide on a per-log-stream-base whether to use the CF or DASD-only for logging.
  - If log streams are to be merged from multiple z/OS systems, you have to use CF.
  - Maybe you use DASD-only logging for stand-alone or test systems.

Log Streams Required by CICS TS

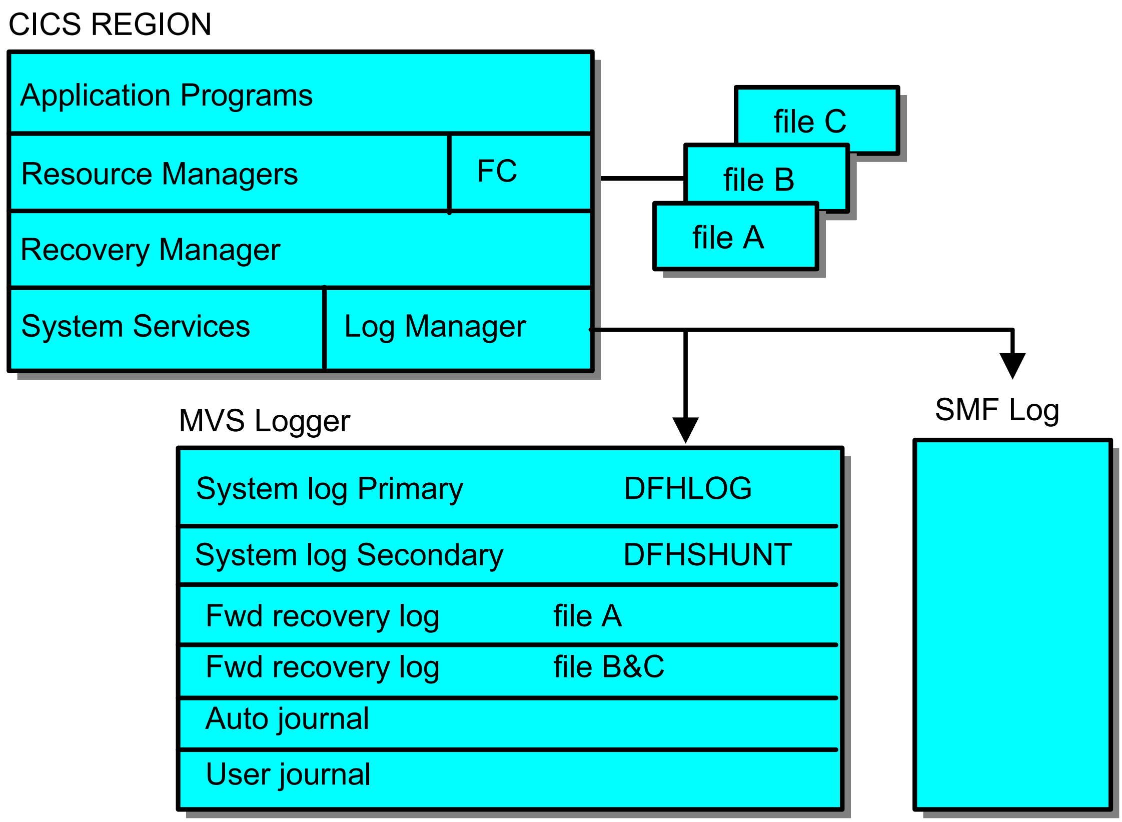


Figure 12-29. Log Streams Required by CICS TS CI207.0

Notes:

This visual shows log streams for a single CICS TS region.

- As already introduced, the System Log is implemented as two log streams with fixed names:
  - The primary system log, **DFHLOG**, is designed to contain recovery records (before images) of active units of work.
  - The secondary system log, **DFHSHUNT**, contains recovery records of long-running units of work and of distributed UOW for which commit processing could not be completed with one of the partner systems.
- All other logs supported by CICS are implemented as log streams:
  - Forward recovery logs for file recovery
  - Journals for auto-journaling used by File Control and Terminal Control
  - User journals for access through the `EXEC CICS WRITE JOURNALNAME` command.
- These three types of logs may be merged onto the same log stream, whereas the system log streams are always separate from all others.

Log Streams for Multiple CICS TS Regions

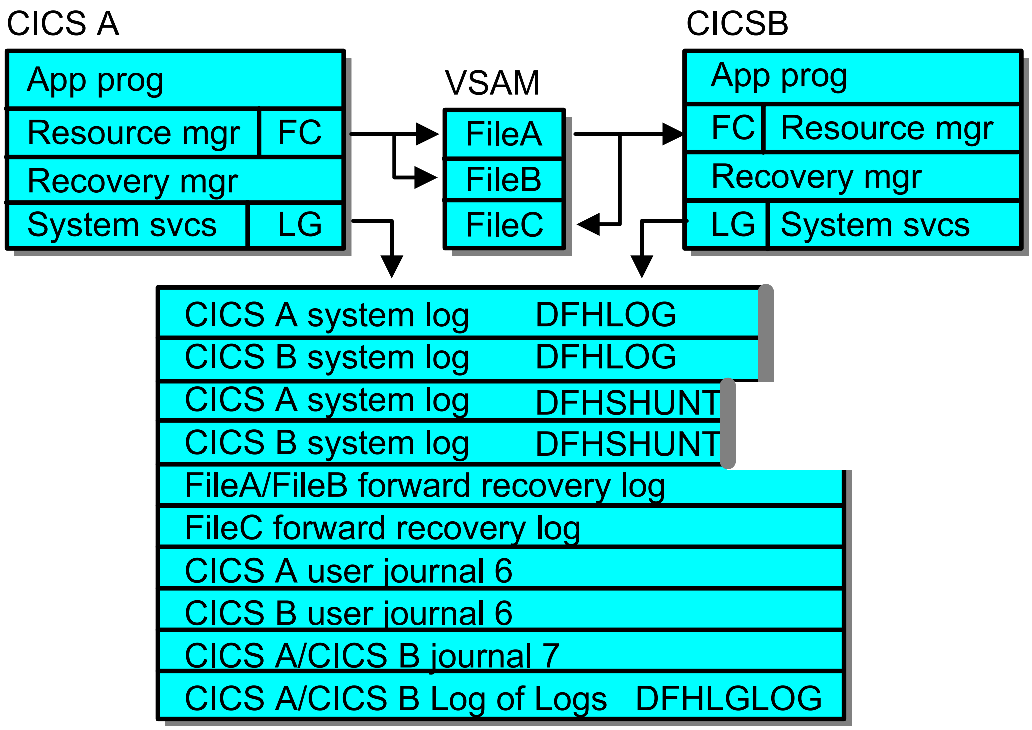


Figure 12-30. Log Streams for Multiple CICS TS Regions CI207.0

Notes:

The following statements are true when a coupling facility is used for logging. For DASD-only log streams, the share level can extend only to the z/OS system.

- Each CICS TS region must have its own distinct system log streams, `DFHLOG` and `DFHSHUNT`.
- Forward recovery log streams may be shared across all CICS regions within the sysplex.
- User journals can be unique to a specific CICS, or can be shared with other CICS TS regions within the sysplex.
- `DFHLGLOG` is the **Log of Logs** used by CICSVR (and maybe other products performing VSAM recovery). It should be a single log stream shared by all CICS regions that use the same recoverable resources.
- All these log streams may be contained within a single CF structure, or use multiple structures.
- It is recommended to use separate structures for test and production systems.

The Log Stream Definition Set

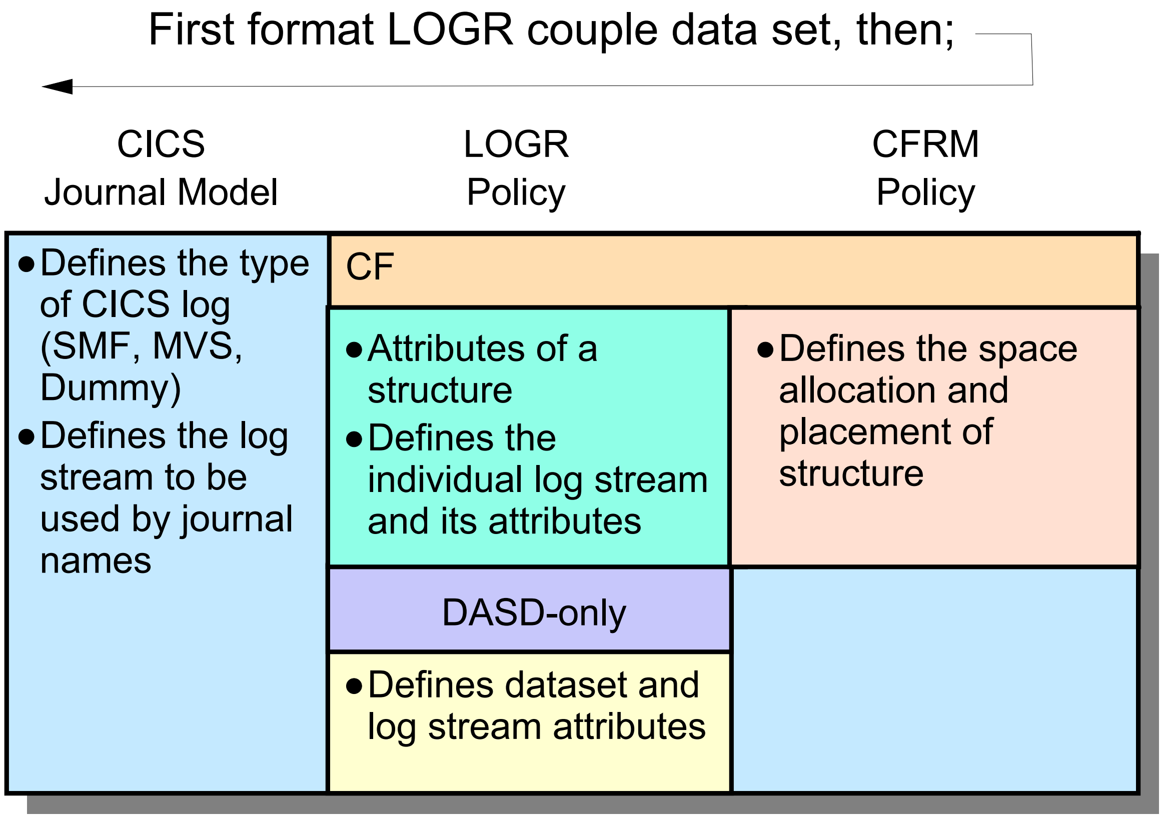


Figure 12-31. The Log Stream Definition Set CI207.0

Notes:

Setting up the log stream environment for a CICS region requires definition efforts on different levels, by means of different utilities. You have to start in the z/OS area, then provide specifications that take effect in the logger environment, and finally activate the appropriate CICS definitions.

Log Stream Definition Overview (CF Logs)

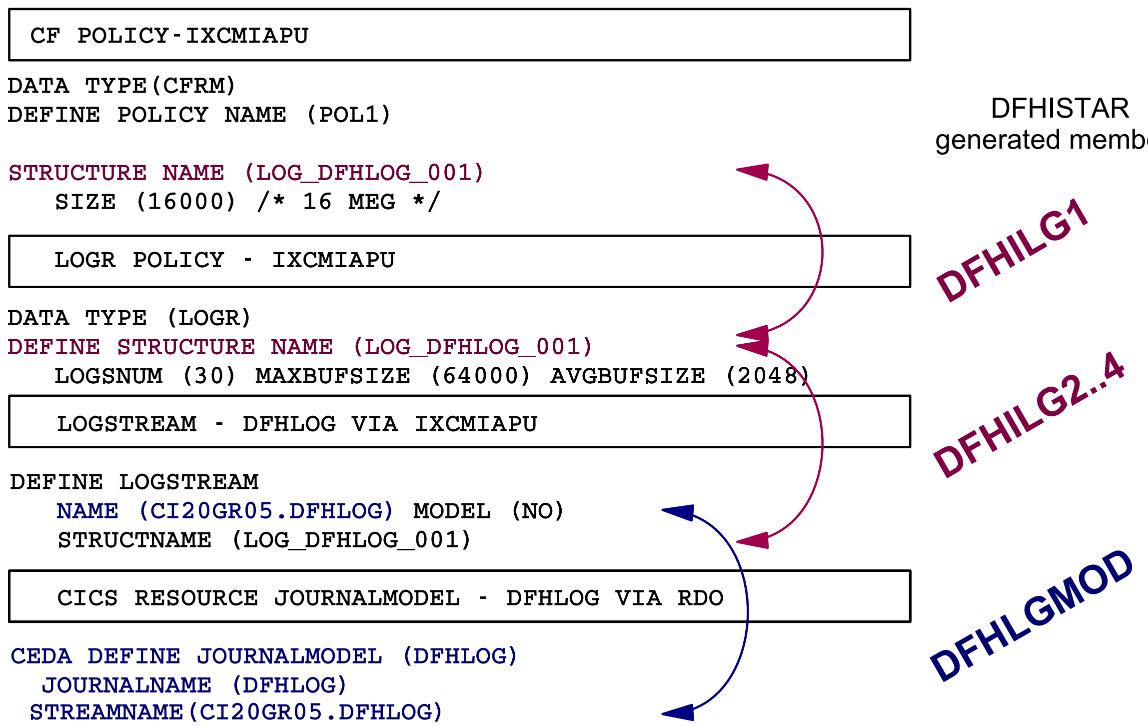


Figure 12-32. Log Stream Definition Overview (CF Logs) CI207.0

Notes:

- `IXCMIAPU` is a z/OS utility that supports definition and maintenance of Coupling Facility policies. It is normally loaded from `SYS1.MIGLIB`.
- **CF Structures** have to be defined *with the same name* to the Coupling Facility in the **CFRM policy** as a list structure and to the z/OS Logger in the **LOGR policy**.
- **Log streams** are defined to the z/OS Logger and refer to a structure.
  The objects defined may be real log streams or **model** log streams that are used by the logger to dynamically create real log streams. More about this later.
- Within CICS, the journal name used by CICS TS as a system or by application programs is resolved by a `JOURNALMODEL` RDO definition that contains this journal name as a full or a generic name, and provides the name of the log stream that implements this journal.
- JCL created by the DFHISTAR process can be used to create the policies and log stream definitions. A standard set of matching JOURNALMODELs is contained in IBM-supplied RDO group DFHLGMOD.

Log Structure Naming and Sizing Suggestions

- `LOG_DFHLOG_001` - Primary System Log Structure
  - Large CF structure will ensure that data set does not spill to DASD
  - Average buffer size tends to be small
- `LOG_DFHSHUNT_001` - Secondary System Log Structure
  - Infrequently used, hence small CF structure
  - Average buffer size tends to be small
- `LOG_USERJRNL_001` - User Journal and Forward Recovery Log Structure
  - Small CF structure, allow for DASD spill and migrating
  - Audit and forward recovery cycle

Figure 12-33. Log Structure Naming and Sizing Suggestions CI207.0

Notes:

- The structure name is specified in the CF CFRM and LOGR policy definitions and referenced by the `STRUCTNAME` parameter of the log stream definition.
- There are few restrictions on the names, but you are recommended to adopt a convention that helps to identify what the structure is used for.
- The names shown above are used by the CICS TS manuals and samples.
- CICS TS provides a utility named **DFHLSCU** that helps to determine the sizes of coupling facility structures. Its use is documented in the *CICS Operations and Utilities Guide, SC34-6431*, chapter 6, and the *CICS Transaction Server for z/OS Version 3.1 Installation Guide, GC34-6426*, chapter 2.
- The structure names shown are the ones used by the jobs created by the CICS TS installation job generator (DFHISTAR).

Log Stream Definition Overview for DASD Logging

```
LOGSTREAM - DFHLOG VIA IXCMIAPU
DATA TYPE (LOGR)

DEFINE LOGSTREAM
   NAME (CI20GR05.DFHLOG) MODEL (NO)
   DASDONLY (YES)
   MAXBUFSIZE (64000)

DEFINE LOGSTREAM
   NAME (CI20GR05.DFHSHUNT) MODEL (NO)
   DASDONLY (YES)
   MAXBUFSIZE (64000)

CICS RESOURCE JOURNALMODEL - DFHLOG VIA RDO
CEDA DEFINE JOURNALMODEL (DFHLOG)
   JOURNALNAME (DFHLOG)
   STREAMNAME (CI20GR05.DFHLOG)

CEDA DEFINE JOURNALMODEL (DFHSHUNT)
   JOURNALNAME (DFHSHUNT)
   STREAMNAME (CI20GR05.DFHSHUNT)
```

DFHISTAR generated members

DFHILG5,6 (modified)

DFHLGMOD (modified)

Figure 12-34. Log Stream Definition Overview for DASD Logging CI207.0

Notes:

You also use the IXCMIAPU utility to define DASD-only logs to the z/OS logger.

- There are, of course, no CF policies and CF structures to be defined for DASD-only logs.
- DASD-only log streams are characterized by setting the IXCMIAPU parameter `DASDONLY` to `YES`.
- Again, JCL with sample definition input can be created by the DFHISTAR process.
- The method of physical logging is *not* visible from the CICS JOURNALMODEL definition. CICS just has to identify the name of the log stream to be used.

Characteristics of Log Streams

```
//LOGDEF   JOB ....
//DEFINE   EXEC  PGM=IXCMIAPU
//SYSIN DD*
DATA TYPE(LOGR)

DEFINE LOGSTREAM
  NAME(MVS1.DFHLOG.MODEL) MODEL(YES)
  STRUCTNAME(LOG_DFHLOG_001)
  DASDONLY(YES)HLQ(CICSP) MAXBUFSIZE(65532)
  LS_DATACLAS() LS_STORCLASS() LS_SIZE()
  STG_DATACLAS() STG_STORCLASS() STG_SIZE()
  HIGHOFFLOAD(80) LOWOFFLOAD(40)
  RETPD(0) AUTODELETE(NO)

DEFINE LOGSTREAM
   NAME(CI20GR05.DFHLOG)
   LIKE(MVS1.DFHLOG.MODEL)
```


Figure 12-35. Characteristics of Log Streams CI207.0

Notes:

- The log stream **NAME** may be a dotted name, up to 26 characters in length. It is required here. Log stream names automatically determined by CICS are of the type `Userid.Applid.Journalname`.
- For DASD log streams, the **DASDONLY** parameter has to be set to YES; otherwise, for CF log streams, it defaults to NO.
- **STRUCTNAME** identifies the CF list structure to which all log blocks of this log stream will be written, and therefore is valid for CF log streams only.
- For DASD log streams, the **HLQ** parameter identifies the prefix which is to be used as a high-level qualifier for the data set names. This is not part of the log stream name. The default is IXGLOGR.
- Set **HIGHOFFLOAD** to a value between 75 and 85 (percent), in order to avoid the log stream filling up completely before offload completes.
  LOWOFFLOAD should be in the range of 40 to 60% for DFHLOG and DFHSHUNT. For user journals and forward recovery logs, specify LOWOFFLOAD as 0.

- The **LS_**CLAS** parameters specify SMS management and storage classes for the CF log stream allocation, while **LS_SIZE** specifies their size, in 4K blocks.
- **STG_**CLAS** are the appropriate specification for the DASD staging data sets, or the DASD-only log data sets, respectively.
- Using the **STG_DUPLEX** and **DUPLEXMODE** parameters, you specify the conditions under which the CF log data is to be duplexed in DASD staging data sets (only applicable to CF log streams).
- **RETPD** (a time interval) and **AUTODELETE** (yes/no) control how long log data is to be kept in the log stream. Possible combinations are:
  - RETPD(0) AUTODELETE(NO)
    The log data is eligible for deletion immediately, but won't actually go away until its deletion is explicitly requested by a call to the z/OS logger.
    This is the recommended setting for DFHLOG and DFHSHUNT, where the CICS logger domain controls the real deletion of the data.

  - RETPD(>0) AUTODELETE(NO)
    The log data is eligible for deletion at the expiration of the retention period (RETPD), but won't actually go away until its deletion is explicitly requested by a call to the z/OS logger.

  - RETPD(0) AUTODELETE(YES)
    The data goes away automatically with no user intervention when the next log data set is allocated.

  - RETPD(>0) AUTODELETE(YES)
    The data goes away automatically, *either* when all data has been marked deletable, *or* at the expiration of the RETPD.
    This might be a suitable option for forward recovery logs.

- The definition may be a **MODEL** for real log stream definition objects that refer to it via the **LIKE** parameter.

JOURNALMODEL Resource Definition

- Provides connection between CICS journalname and associated MVS logstream or SMF log

```
Journalmodel ==> ........
Group        ==> ........
Description  ==> ......................
Journalname  ==> ........  name|JOURNALMODEL name
Type         ==> MVS        MVS | SMF | DUMMY
Streamname   ==> stream_name_template|
                                &USERID..&APPLID..&JNAME
```


Figure 12-36. JOURNALMODEL Resource Definition CI207.0

Notes:

- The **JOURNALNAME** can be coded as a specific name, or generically, containing "**%**" and "*" for wildcard characters or strings.
  If the **JOURNALNAME** parameter is omitted, it defaults to the name of the JOURNALMODEL resource name.
- **STREAMNAME** can specify an explicit log stream name, or a template used to construct the log stream name by use of the variables shown. This field is applicable only to journal models defined as TYPE(MVS).
  - The stream name defaults to `USERID.APPLID.JNAME`, where `USERID` is the CICS Region userid, `APPLID` is the name specified in the appropriate SIT parameter, and `JNAME` is the journal name, as specified by the log accessing instance.
  - This default naming convention provides a unique log stream name for every journal.
- **TYPE(DUMMY)** means that logging requests by CICS instances will be handled and answered to as if anything went fine, but actually no logging is performed.
  If CICS system logs are specified as TYPE(DUMMY), there is no restart information available, and CICS TS may only perform INITIAL starts.

Journal Names

- For CICS system logs, fix names are used
  - **DFHLOG** - Primary system log
  - **DFHSHUNT** - Secondary system log
  - CICS supplied JMODEL definitions in RDO group DFHLGMOD
  - Type may be MVS or DUMMY
- For auto-journals and forward-recovery logs (non-RLS)
  - Journal names must be DFHJnn, where nn=01..99
  - Auto journals can be MVS or DUMMY or SMF
  - Forward-recovery logs can be MVS or DUMMY
  - Names in JMODEL definition can be specific or generic
- User journals
  - Can use any journal name, up to eight characters
  - Can be MVS or DUMMY or SMF
  - Names in JMODEL definition can be specific or generic

Figure 12-37. Journal Names CI207.0

Notes:

- Note that the JOURNALMODELs that determine the log stream names for the CICS system logs may be named anyway. However, for ease of use and clarity, it is highly recommended to use the given journal names.
- User journals, auto journals, and forward recovery logs from a single CICS region or multiple CICS regions within a sysplex may share log streams.
  If they are defined as using the same log stream, data from multiple journals or logs in each CICS region may be merged onto one log stream.
- For **non-RLS VSAM files**, the forward recovery log stream may be defined in two ways:
  - Directly in the ICF catalog
  - Through CICS `FILE` and `JOURNALMODEL` definition
- For **VSAM RLS files**, forward recovery log streams can only be specified in the VSAM ICF catalog.

JOURNALMODEL Examples

```
DEFINE JOURNALMODEL(DFHLOG)  GROUP(LOGS)
          JOURNALNAME(DFHLOG)
          STREAMNAME(CI20GR05.DFHLOG)
          TYPE(MVS)
DEFINE JOURNALMODEL(DFHJ08)  GROUP(LOGS)
          JOURNALNAME(DFHJ08)
          TYPE(SMF)
DEFINE JOURNALMODEL(USERJNL9)  GROUP(LOGS)
          JOURNALNAME(DFHJ09)
          TYPE(DUMMY)
DEFINE JOURNALMODEL(UJRNLX)  GROUP(LOGS)
          JOURNALNAME(DFHJ1%)
          STREAMNAME(&USERID.MERGED.USRJRNLS)
          TYPE(MVS)
DEFINE JOURNALMODEL(OTHERS)  GROUP(LOGS)
          JOURNALNAME()
          STREAMNAME(&USERID..&APPLID..&JNAME)
          TYPE(MVS)
```


Figure 12-38. JOURNALMODEL Examples CI207.0

Notes:

Given the following definitions for a CICS region executing with userid `CIMAIN` and with `APPLID=CICSAGT1` coded in the SIT, records written:

- To the CICS system log for this region will be written to log stream `CI20GR05.DFHLOG`.
- To auto journal “08” or user journal “08” using an `EXEC CICS WRITE JOURNALNAME(DFHJ08)...` or `EXEC CICS WRITE JOURNALNUM(DFHJ08)...` command will be written in SMF format to the MVS SMF log stream.
- To auto/user journal “09” using an `EXEC CICS WRITE JOURNALNAME(DFHJ09)...` or `EXEC CICS WRITE JOURNALNUM(DFHJ09)...` command will not be written to any log stream, though the application program will still receive a normal response.
- To user journals 10-19 (DFHJ10-DFHJ19) will be merged on log stream `CIMAIN.CICSAGT1.USRJRNLS`, together with records from any other CICS regions running under the same userid and with the same JOURNALMODELs installed.

All other forms of journaling will use `JOURNALMODEL(OTHERS)`, resolving the log stream name to the default name that CICS would use if there was no matching `JOURNALMODEL` entry for a journal name.

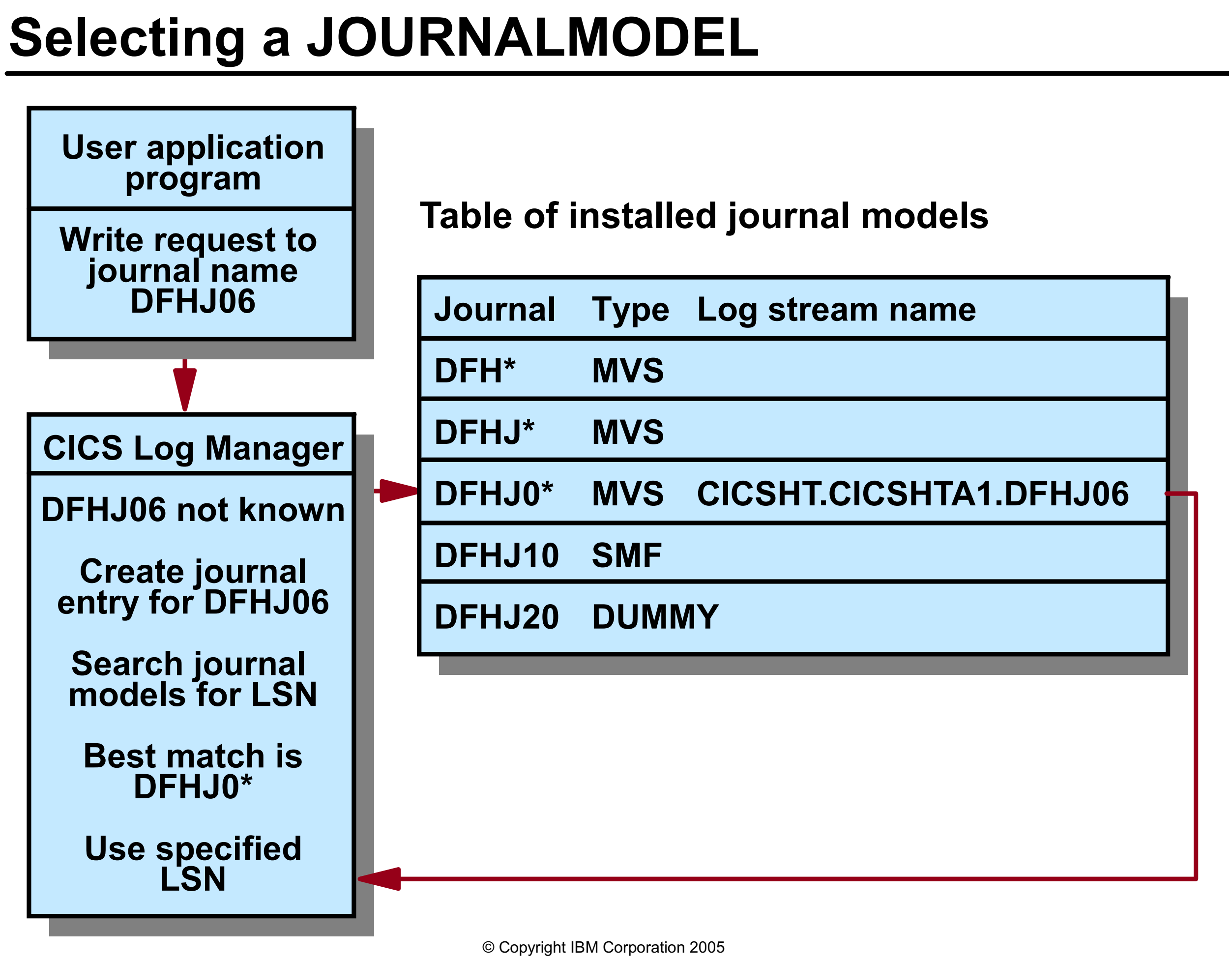


Figure 12-39. Selecting a JOURNALMODEL CI207.0

Notes:

- When CICS is about to access (normally write to) a log or journal for the first time, both for system purposes and on servicing an application request, it has to determine two things:

  a. The JOURNALMODEL to be selected, based on the given journal name and the active (installed) JOURNALMODEL definitions

  b. The log stream name (LSN) to be connected to, which is identified by the JOURNALMODEL definition

  Assume a userid and an applid that is used by CICS, and create your own examples for selecting the JOURNALMODEL and determining the name of the log stream (LSN).

Resolving the System Log Stream and Dynamic Creation of Log Streams

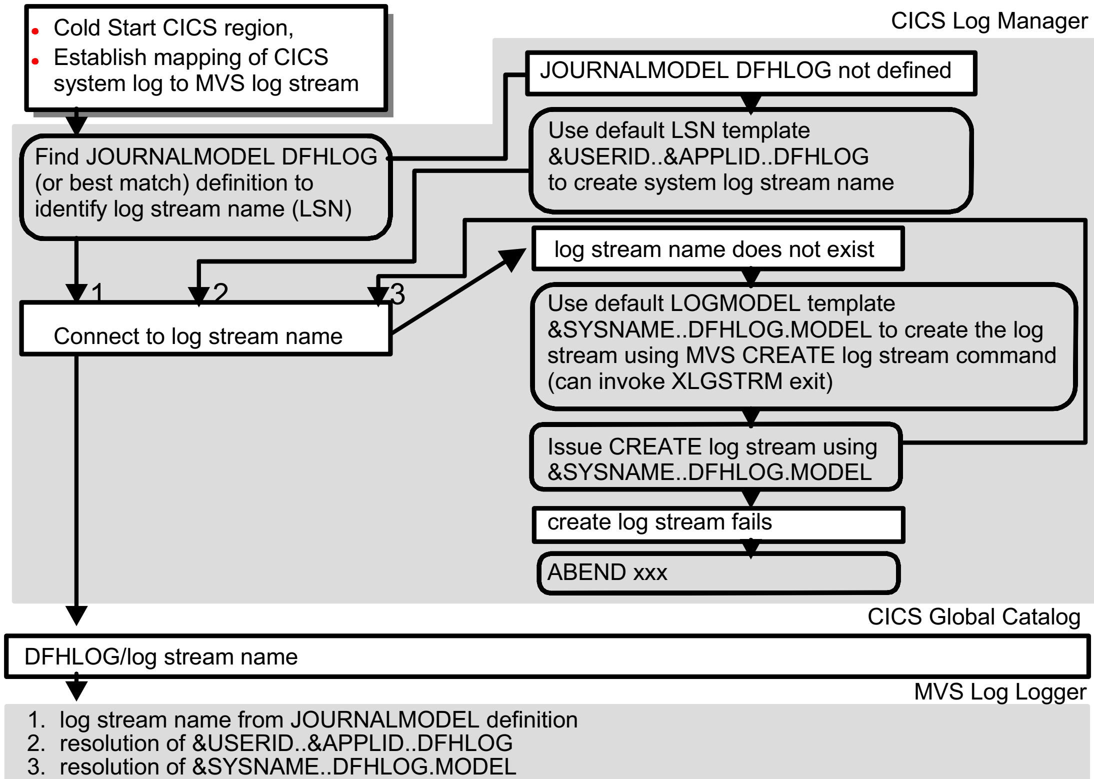


Figure 12-40. Resolving the System Log Stream and Dynamic Creation of Log Streams CI207.0

Notes:

- Remember that the resolution of a template LSN specified in the JOURNALMODEL resource definition is done by CICS, so CICS always uses a real name when attempting to connect to the log stream.
- If the log stream named by CICS does not exist, CICS tries to dynamically create it using the MVS `CREATE LOG STREAM` command and referencing a MODEL log stream named by the following rules:
  - For **system log streams**, in other words, for DFHLOG and DFHSHUNT, the `CREATE` request will be based on model **`&SYSNAME..DFHLOG.MODEL`**, where SYSNAME is the name of the z/OS system. If the define fails, CICS abends.
  - The LSN identified in the JOURNALMODEL must not be the name of a model!

Resolving a User Journal Name

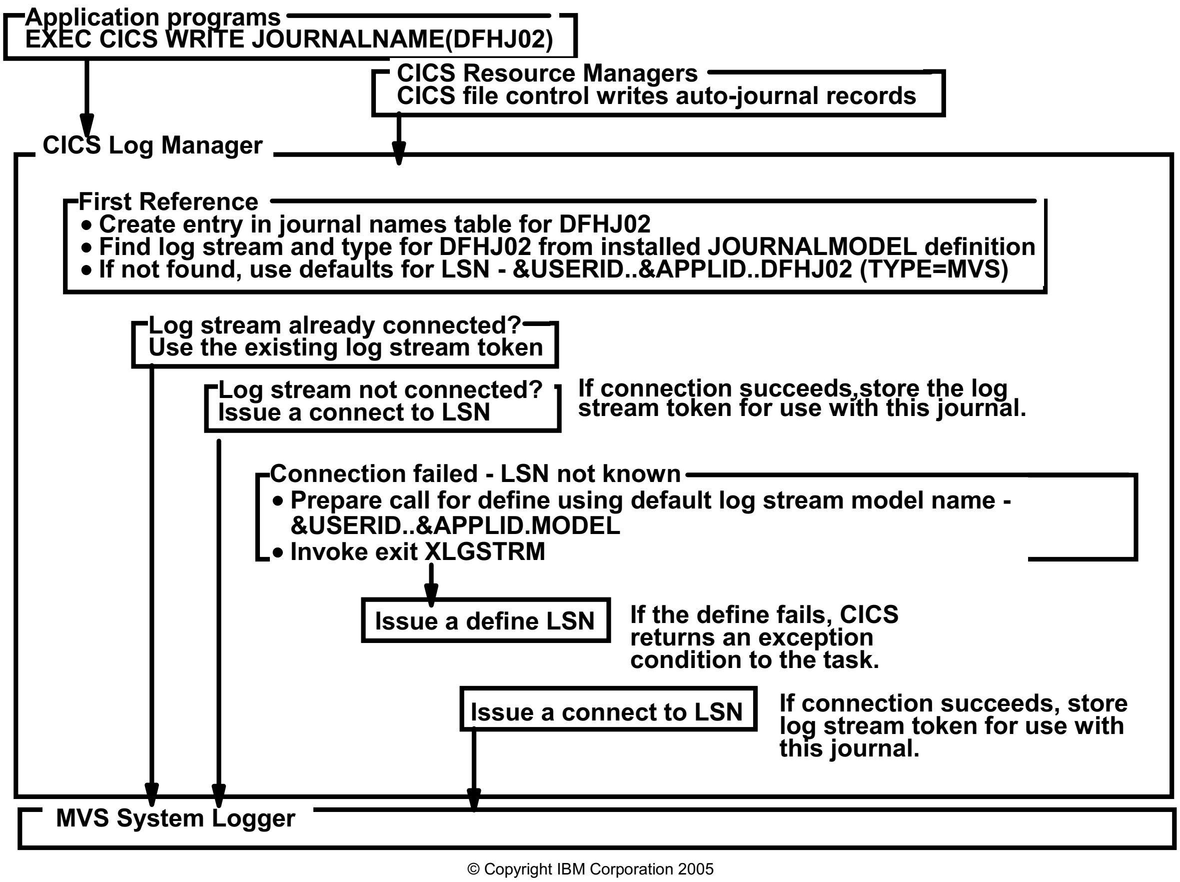


Figure 12-41. Resolving a User Journal Name CI207.0

Notes:

Within the z/OS logger, there is no difference between the handling of CICS system log streams and non-system log streams, as the logger is "not aware" of any differences.

What is different is the model name, which is constructed and used by CICS in order to request the dynamic creation of a log stream which does not exist yet:

- The model name for non-system journals or logs is based on the log stream name, with just the last qualifier changed to `MODEL`.
  So, if the original request is for a log stream named `CIMAIN.CICSHTA1.DFHJ99`, and this is not there, the model name used on the `CREATE LOG STREAM` is `CIMAIN.CICSHTA1.MODEL`.
  If the define fails, CICS returns an exception condition to the task.

Handling JOURNALMODELs

```
CEDA INSTALL  JOURNALMODEL

CEMT INQUIRE JOURNALMODEL
             JOURNALNAME

CEMT INQUIRE STREAMNAME

CEMT SET     JOURNALMODEL

CEMT DISCARD JOURNALMODEL
             JOURNALNAME
```


Figure 12-42. Handling JOURNALMODELs CI207.0

Notes:

- JOURNALMODELs have to be installed to become active, as with any other resource definitions, and may be deactivated and removed by CEMT DISCARD.
- When CICS has processed a journal request successfully, the appropriate **journal name** is added to an internal journal names table and to the global catalog.
  - The status of these logs can be inquired on using CEMT INQUIRE JOURNALNAME. This command also returns the name of the log stream used for the named journal.
  - CEMT DISCARD JOURNALNAME removes journal name entries from the journal name table and the catalog.
  - CEMT SET JOURNALNAME allows you to enable and disable a CICS user journal.
- Information about currently connected log streams may be obtained by using the CEMT INQUIRE STREAMNAME command.
- *Attention*: When you change a JOURNALMODELs definition, this will not take effect unless you discard the JOURNALNAME entry based on the original JM definition.
- All CEMT commands are available as EXEC CICS SPI commands.

12.5 Defining Recoverable Resources

CICS Resources Eligible for Recovery

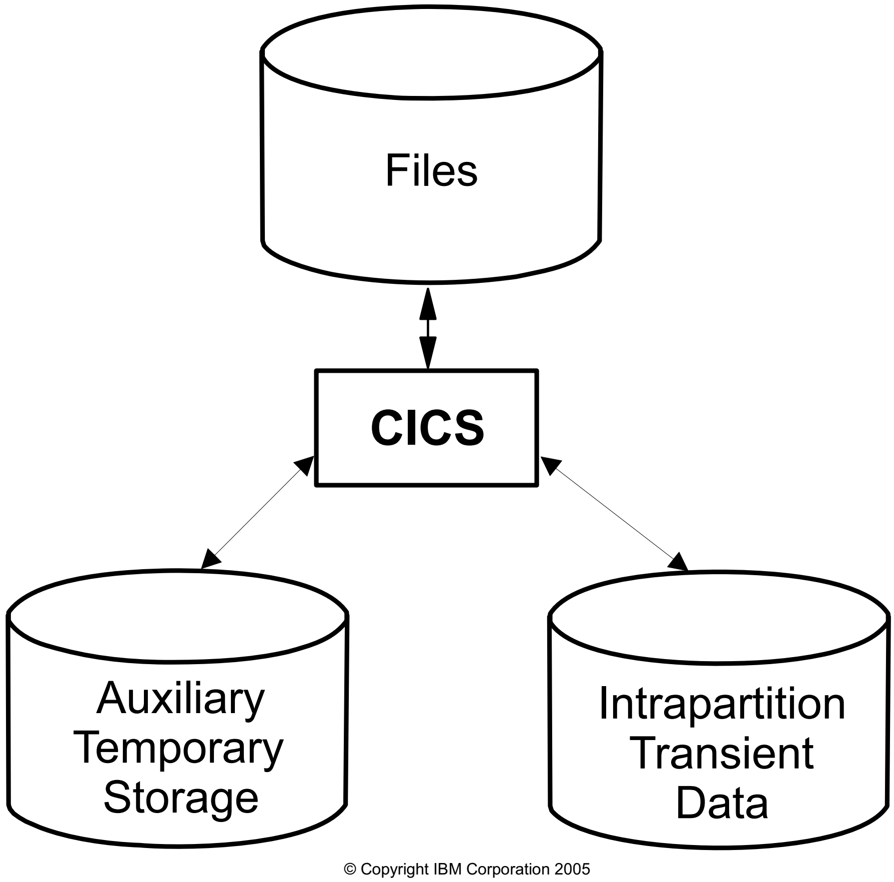


Figure 12-43. CICS Resources Eligible for Recovery CI207.0

Notes:

In CICS, three resource types may be defined as recoverable:

- File
- Auxiliary Temporary Storage
- Intrapartiton Transient Data

Specifying Recoverable Files (Non-RLS)

Pure backout support:

```
CEDA   DEF FILE (FILEA)
  ...
  RECOVERY(BACKOUTONLY)
  ...
```

Backout and forward recovery support:

```
CEDA   DEF FILE (FILEA)
  ...
  RECOVERY(ALL)
  FWDRECOVERYLOG(nn)
  ...
```

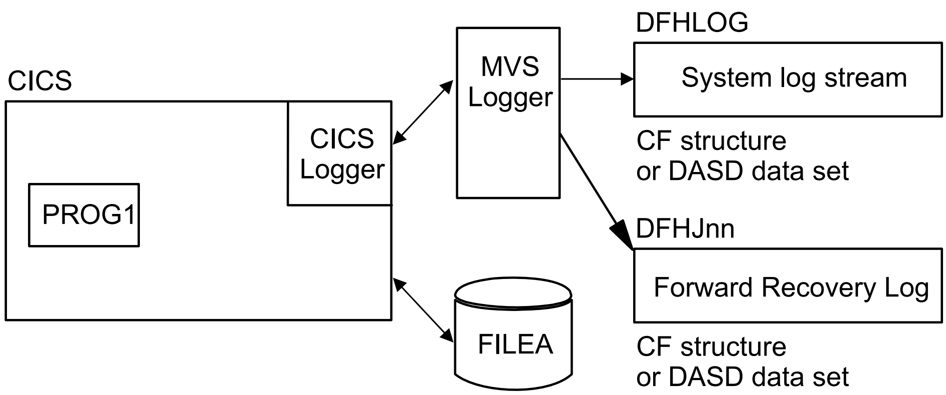


Figure 12-44. Specifying Recoverable Files (Non-RLS) CI207.0

Notes:

- The file definition option RECOVERY(BACKOUTONLY) defines a file FILEA as a recoverable data set, in the sense in which the term is usually used.
  - CICS extended exclusive control will be used.
  - “Before images” are written to the *system log* (log stream) for backout in the event of a task or a system failure.
- The file definition option RECOVERY(ALL) defines a file FILEA as a recoverable data set, too, additionally requesting forward recovery support.
  In addition to the system log records written for backout:
  - “After images” of changed records are written to a specific **forward recovery log** (-stream), the journal name of which has to be specified by the two-digit numeric FWDRECOVLOG parameter.
  - The forward recovery log resultant journal name is always **DFHJ** plus the two-digit suffix.

- CICS VSAM Recovery Version 3.3 (CICSVR for OS/390 and z/OS, program product 5655-H91), includes support for these types of records, as well as those created if you specify `RECOVERY(BACKOUTONLY)`.
- With this method of recovery, there is also the option to take backups while the file is open for update by CICS. This is sometimes called “fuzzy” backup, and it requires `FWDRECOVLOG(n)` and a utility like CICSVR. To allow this, specify `BACKUPTYPE(DYNAMIC)`.

Automatic Journaling

```
CEDA   DEF FILE (FILEA)
  AUTO JOURNALLING
   JOurnal      ==> No | 1 - 99
   JNLRead      ==> None|Upd|Read|All
   JNLSYNCRead  ==> No | Yes
   JNLUpdate    ==> No | Yes
   JNLAdd       ==>
None|Before|After|All
   JNLSYNCWrite ==> Yes | No
```

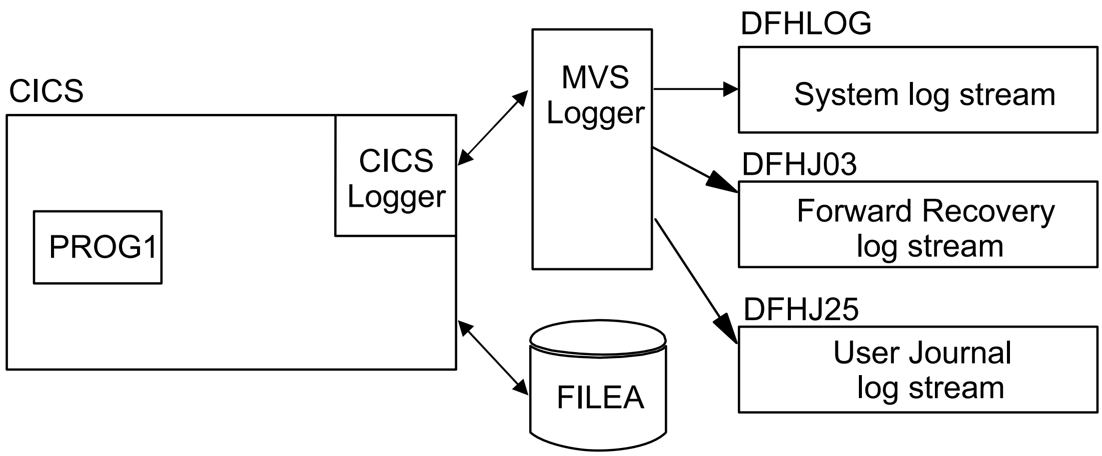


Figure 12-45. Automatic Journaling CI207.0

Notes:

- **Supplementary to** the backout and forward recovery options, journaling for distinct file operations may be requested by file definition.
- `JNLADD(BEFORE)` specifies that the record is written to the journal before the VSAM I/O operation.
- Recovery utilities written for pre-ESA releases of CICS may require these journal records, as do any utilities that may be used for auditing or review purposes.
- Note that this is just a journaling service that CICS offers. CICS just writes the journal data, as requested in this file definition section, but will never access this data itself thereafter.
  So, it is completely up to the installation to handle the appropriate log stream data.

DFHCSD Recovery

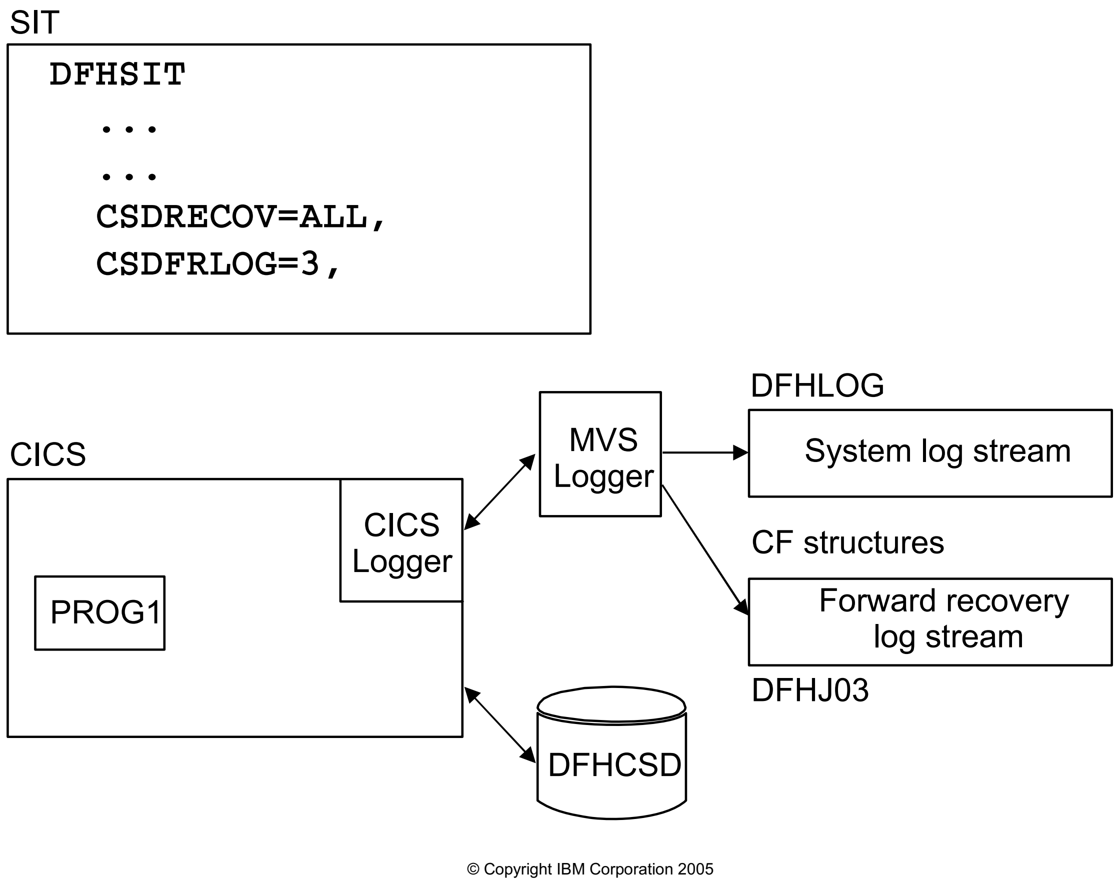


Figure 12-46. DFHCSD Recovery CI207.0

Notes:

- Since the CSD file contains resource definitions, you define it in the SIT, along with its recovery options. The same recovery options are available as for RDO-defined files.
- The standard (CICS-supplied) SIT (`DFHSIT6$`) specifies `CSDRECOV=ALL` and `CSDFRLOG=1`.
  So, a JOURNALMODEL that specifies a log stream name to be associated to the DFHJ01 journal name must be active, unless the default rules are to take effect, which would result in a log stream named **`&userid..&applid.DFHJ01`**.
- Automatic journaling is supported for the DFHCSD file too, and controlled by the SIT parameter `CSDJID=number`. Mapping to a log stream works in the same way as `CSDFRLOG`.
  The automatic journaling options enforced for the CSD when you code `CSDJID=number` are `JNLADD=BEFORE` and `JNLUPDATE=YES`. These options are sufficient to record enough information for a user-written forward recovery utility. No other automatic journaling options are available for the CSD.

Specifying Recoverable TD and TS Queues

```
CEDA  DEFine TDqueue( TDWT )
 TDqueue       ==> TDWT
 ...
INTRA PARTITION PARAMETERS
 Atifacility   ==> ...
 RECOVstatus   ==> Logical
 Facilityid    ==> ...
```

```
CEDA  DEFine TSmodel( RECTS)
 TSmodel           : RECTS
  ...
 PRefix         ==> REC
 Location       ==> Auxiliary
RECOVERY ATTRIBUTES
 RECovery       ==> Yes
```

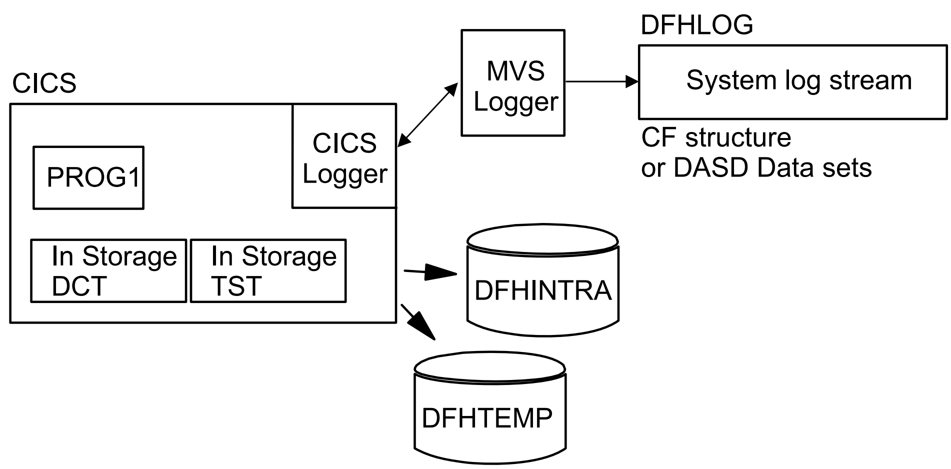


Figure 12-47. Specifying Recoverable TD and TS Queues CI207.0

Notes:

Backout recovery may be requested for INTRApartition Transient Data queues and for Auxiliary Temporary Storage queues.

- Enqueuing support for concurrent accesses and backout support in the event of task or system failures, similar to VSAM files, is requested by:
  - Defining TD queues as **logically recoverable**
  - Defining TS queues as **recoverable**
- Instead of writing queue data to the system log, CICS performs updates to the queues immediately, and logs write and read *pointers*. These pointers are also held in the main storage DCT and TST. Task backout is performed by use of the main storage entries. The system log is used only for emergency restart.
- The **Physical** recovery option is available for TD queues only. It takes effect only in the event of an emergency restart. If this occurs, the queue is recovered to its status at the time CICS terminated. No backout support is provided.
- No forward recovery is available for the queuing facilities.

Recovery/Restart Checklist

1. Determine the number, size, and names of the logs you require
2. Set up the Coupling Facility Resources (CFRM and LOGR policy definitions)
3. Define JOURNALMODELS in the CICS CSD
4. Modify SIT:
   - AKPFREQ=
   - START=AUTO
   - TBEXITS=(name1,name2,...,...,...,name6)
5. Specify selected resources as recoverable
   - Files and Transient Date Queues definitions
6. Test changes

Figure 12-48. Recovery/Restart Checklist CI207.0

Notes:

Refer to the *CICS System Definition Guide, SC34-6428*, for examples and details to be observed.

- TBEXITS names up to six user exit programs that can be used during emergency restart. For more information, refer to the *CICS Recovery and Restart Guide, SC34-6246*.
- Never specify more recovery than you plan to use.
- Always practise an emergency restart, to ensure that your definitions and procedures will work when the real thing happens.

Recovery and Restart (Summary)

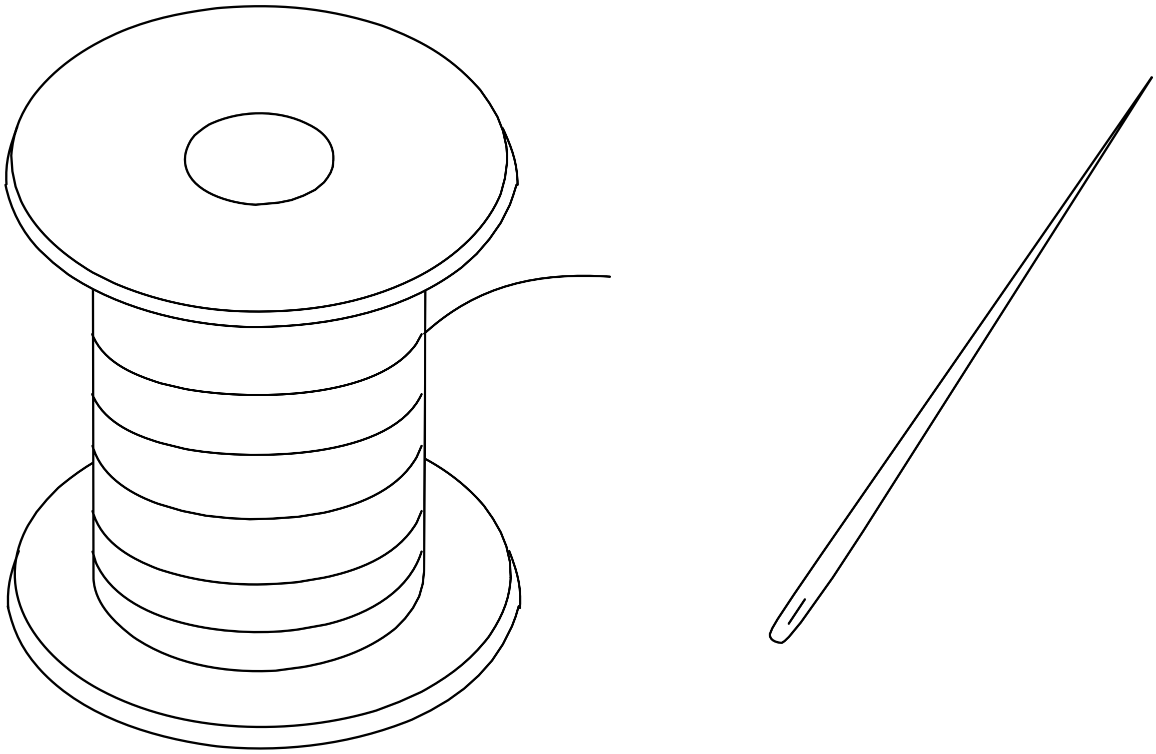


Figure 12-49. Recovery and Restart (Summary) CI207.0

Notes:

- The way CICS restarts depends on the way it terminates.
- Only a normal termination allows control of the process through the shutdown PLT and XLT; then warm start the system.
- All other types of termination require an emergency restart.
- The Warm Keypoint Record saves required information in the Global Catalog Data set for warm restart.
- START=AUTO, the recommended specification, lets CICS determine the correct type of restart. START=INITIAL and START=COLD begin with a clean state ignoring information logged from the previous run, and should never be attempted after an uncontrolled shutdown.
- CICS TS uses the z/OS Logger services to record backout and forward recovery information in separate Coupling Facility log stream structures or DASD-only log streams.
- The resources that are eligible for CICS recovery are:
  - VSAM and BDAM files
  - Auxiliary Temporary Storage and INTRApartition Transient Data queues.

Unit Summary

Having completed this unit, you should be able to:

- Describe the ways of terminating CICS and the corresponding methods of restarting
- Describe the use of the PLT and XLT in shutdown and restart
- State the SIT parameters required for proper recovery and restart of the system
- Describe the purpose of the Activity Keypoint and Warm Keypoint records
- Define the z/OS Coupling Facility structures required for the CICS TS system log and for forward recovery logs
- Identify the options used to define recoverable files and queues

Figure 12-50. Unit Summary CI207.0

Notes:

Unit 13. Security Using an External Security Manager

What This Unit Is About

It is important to protect your CICS resources from unauthorized access. This unit discusses how you can do this using RACF or another suitable external security manager (ESM).

What You Should Be Able to Do

After completing this unit, you should be able to:

- Explain CICS security implementation to the external security manager (ESM) administrator
- State what security definitions are required to run CICS TS under z/OS, and to allow CICS TS to request service from other system components such as VTAM, VSAM, and so on
- Explain and contrast the three basic types of security that may be used
- Define the types of resources that may be protected
- Explain how multiple CICS TS regions accessing the same ESM can be protected independently from each other
- State the means and parameters that support protection of non-terminal-oriented tasks

References

CICS Transaction Server for z/OS Version 3.1 Installation Guide, GC34-6426

CICS System Definition Guide, SC34-6428

CICS Resource Definition Guide, SC34-6430

CICS RACF Security Guide SC34-6011

Unit Objectives

After completing this unit, you should be able to:

- Explain CICS security implementation to the external security manager (ESM) administrator
- State what security definitions are required to run CICS TS under z/OS, and to allow CICS TS to request service from other system components such as VTAM, VSAM, and so on
- Explain and contrast the three basic types of security that may be used
- Define the types of resources that may be protected
- Explain how multiple CICS TS regions accessing the same ESM can be protected independently from each other
- State the means and parameters that support protection of non-terminal-oriented tasks

Figure 13-1. Unit Objectives CI207.0

Notes:

13.1 Protecting CICS Resources

Data Set Protection

```
//CICSA    JOB    ...,USER=CIMAIN,PASSWORD=XYZXYZ
//         EXEC PGM=DFHSIP
//STEPLIB  DD DSN=CICSTS23.SDFHAUTH,DISP=SHR
//DFHRPL   DD DSN=CICSTS23.SDFHLOAD,DISP=SHR
//         DD .... (user libraries) ...
//DFHLCD   DD DSN=SYSPROD.CICSA.DFHLCD,DISP=SHR
// ...     DD ...  (other system data sets) ...
//FILEA    DD DSN=USEPROD.FILEA,DISP=SHR
// ...
```

```
ADDSD    'CICSTS23.SDFH*'   UACC(NONE)
PERMIT   'CICSTS23.SDFH*'   ID(CIMAIN)    ACCESS(READ)

ADDSD    'SYSPROD.CICSA.*'  UACC(NONE)
PERMIT   'SYSPROD.CICSA.*   ID(CIMAIN)    ACCESS(UPDATE)

ADDSD    'USEPROD.FILEA'    UACC(READ)
PERMIT   'USEPROD.FILEA'    ID(CIMAIN)    ACCESS(UPDATE)
```


Figure 13-2. Data Set Protection CI207.0

Notes:

Independent of any CICS resource protection, your installation probably uses RACF (or an equivalent) to protect your data sets.

- The only userid that is checked for data set access is the userid which is associated to the CICS TS job or started task, which is called the **Region Userid**.
- The region userid requires:
  - `READ` access to all installation data sets
  - `UPDATE` access to its system data sets and application file data sets
- If CICS is initiated as a started task, the region userid is defined by a profile in the RACF STARTED class. An alternative, older method of associating started task names and the userid to be used is the RACF started procedures table (ICHRIN03).
- Data set profiles are defined to RACF using the `ADDSD` command. Unique userids, or a group of IDs, may be `PERMIT`ted access the data sets.
- The "CICS Data Set and System Security" chapter of the *CICS RACF Security Guide, SC34-6011*, contains recommendations for access authority for each CICS data set.

Authorizing Access to a VTAM ACB

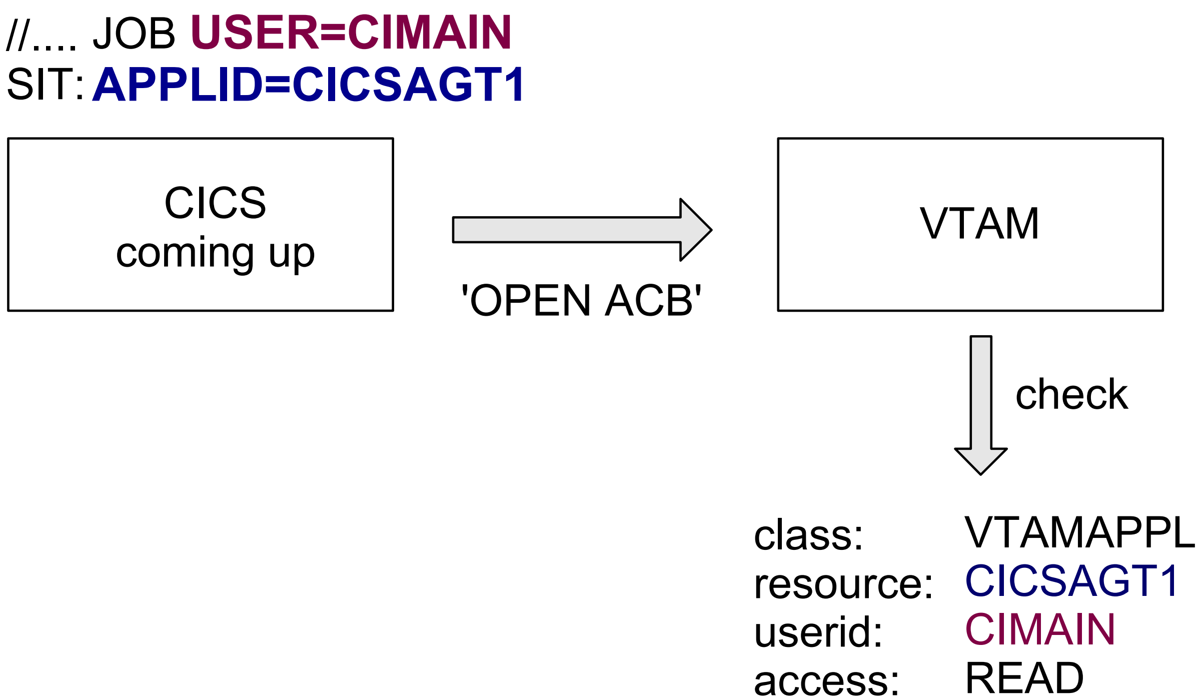


Figure 13-3. Authorizing Access to a VTAM ACB CI207.0

Notes:

- The resource protected here is the **VTAM ACB name**.
- Protection ensures that ACB names defined to VTAM and specified on the `APPLID` SIT parameter can only be used by CICS regions running with authorized userids.
- Profiles are in RACF class **VTAMAPPL**.
- The resource name is the **APPLID** defined in the SIT.
- The userid checked is the CICS Region (Main) Userid which requires **READ** permission for the protected resource.
- These are the RACF commands that define a profile and a permission in this area:

```
PERMIT applid CLASS(VTAMAPPL) ID(userid)
```

- The VTAMAPPL class must be active and RACLISTed for this protection to be in effect.

Authorizing Access to MVS Log Streams

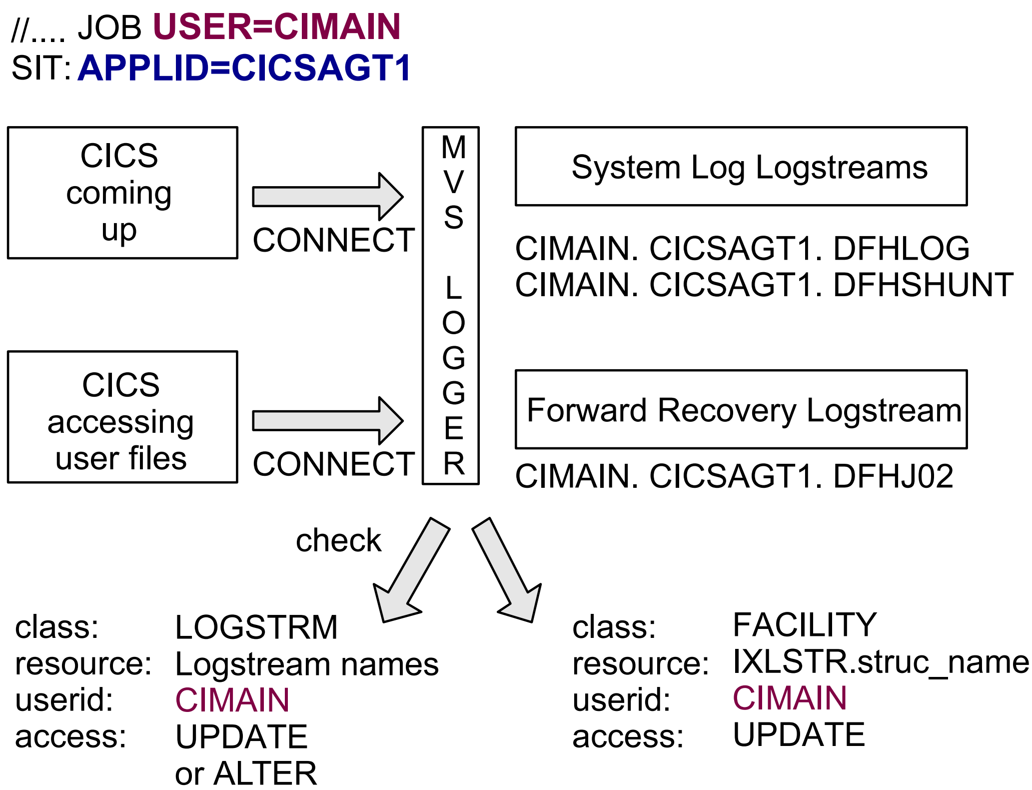


Figure 13-4. Authorizing Access to MVS Log Streams CI207.0

Notes:

- You must ensure that CICS TS regions are authorized to write to (and create, if necessary) the log streams that are used for their system logs and general logs.
- Profiles are in RACF class **LOGSTRM**:
  - The resource name is the name of the log stream to be accessed.
  - The userid checked is the CICS Region Userid.
- A profile and a permission in this area are defined by these RACF commands:

```
RDEFINE LOGSTRM region_userid.applic.* UACC(none)
PERMIT region_userid.applid.* CLASS(LOGSTRM) ID(region_userid)
```

- **UPDATE** authority is required if the log streams are already defined to MVS.
- **ALTER** authority is required if the log streams have to be created dynamically.
  - Additionally, in this case, the CICS Region Userid requires **UPDATE** authority to the relevant coupling facility structures in CLASS(FACILITY) with the profiles named `IXLSTR.structurename`.

13.2 Protecting CICS Resources

Levels of Security

First:
Security to access CICS

Second:
Security to access the transaction

Then:
Security to access resources from that transaction:

- Queues
- Journals
- Programs
- Transactions
- Files
- Commands

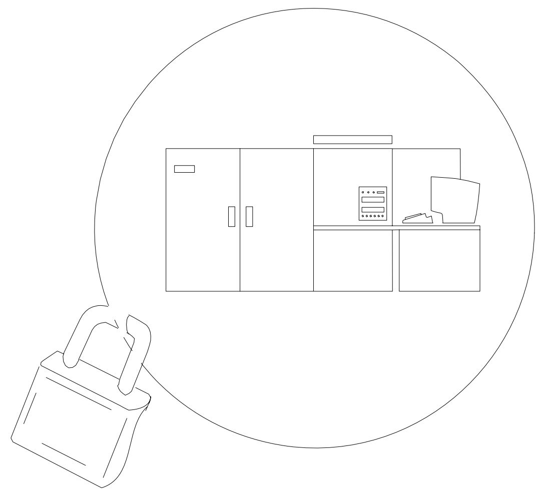


Figure 13-5. Levels of Security CI207.0

Notes:

Design as much protection as your installation requires, but not more:

- Restrict access to the CICS region and to specific terminals
- Protect transactions from unauthorized use
- Prevent some users of a transaction from using certain resources or commands.

Accessing CICS

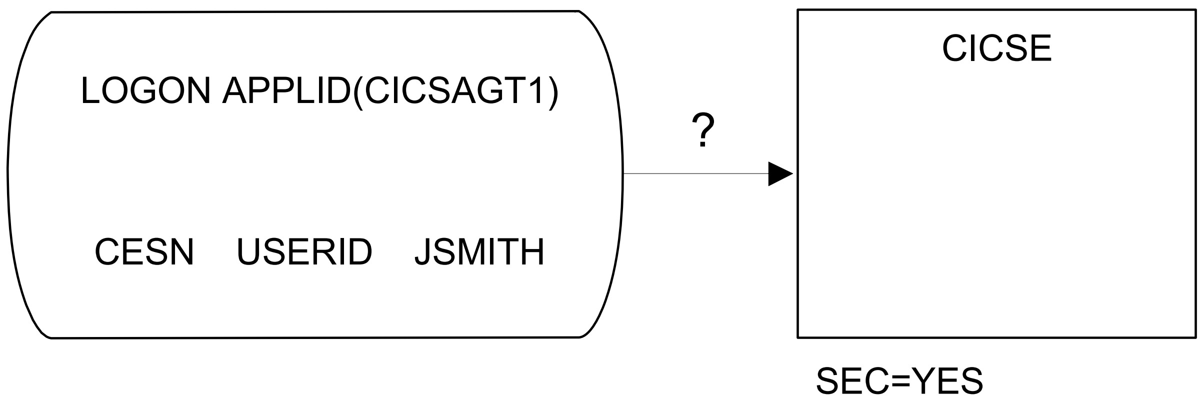


Figure 13-6. Accessing CICS CI207.0

Notes:

- Based on profiles defined in RACF class `APPL`:
  - The CICS region's ACB name has to be defined as a resource of the RACF `APPL` class.
  - The operator's userid (or an appropriate user group) must be permitted READ access to this profile.
- **This checking occurs when a user signs on, if security is active.**
- Access to terminals can be restricted by defining them to RACF in the `TERMINAL` class.

User Security

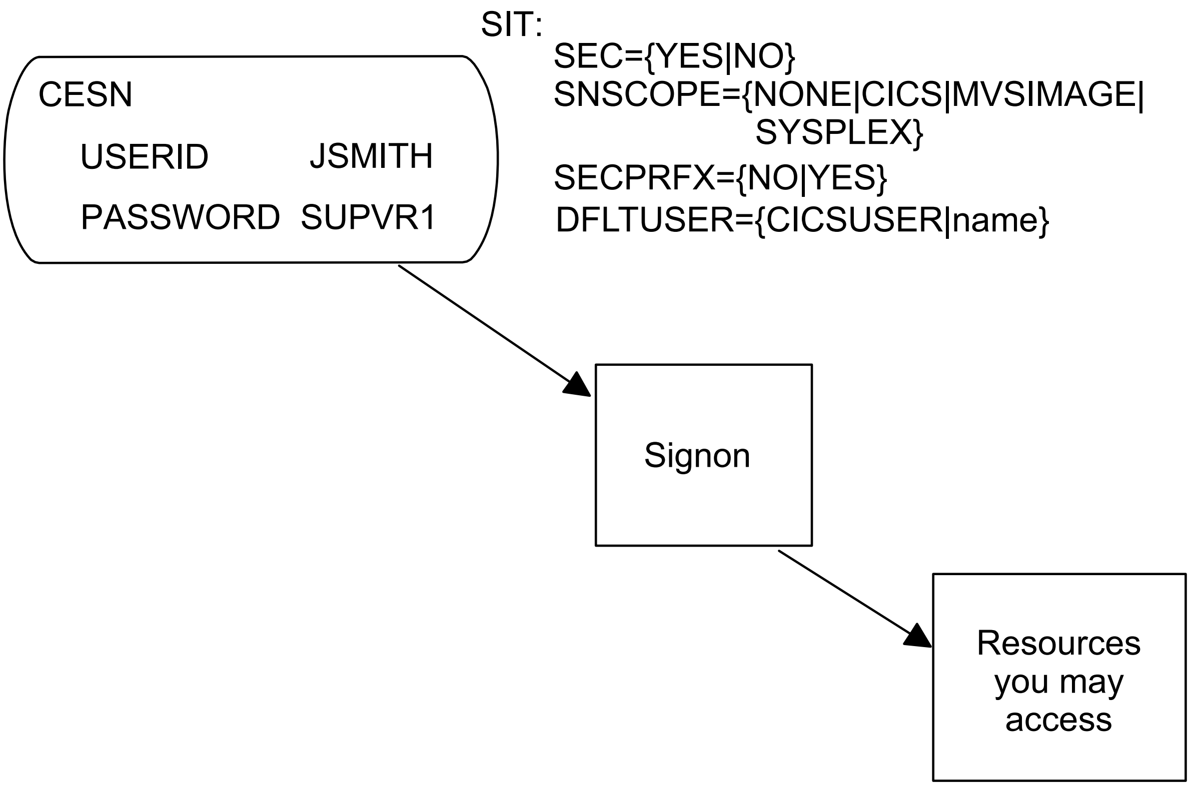


Figure 13-7. User Security CI207.0

Notes:

- If security is active, users must have a security profile before they can access secure resources. There are two ways to accomplish this:
  a. The user signs on using CESN.
  b. An application issues the EXEC CICS SIGNON command.
  c. Before signon, users have the security associated with the default userid you specify in DFLTUSER.
- Define CESN with a PROFILE specifying UCTRAN (YES). This keeps new passwords from being lowercase.
- The CESN screen provides a field for changing passwords.
- User security authorization is always checked against profiles within RACF general resource classes, never against data set profiles.
- The use and meaning of the `SECPRFX` parameter will be discussed later in this unit.

Signon

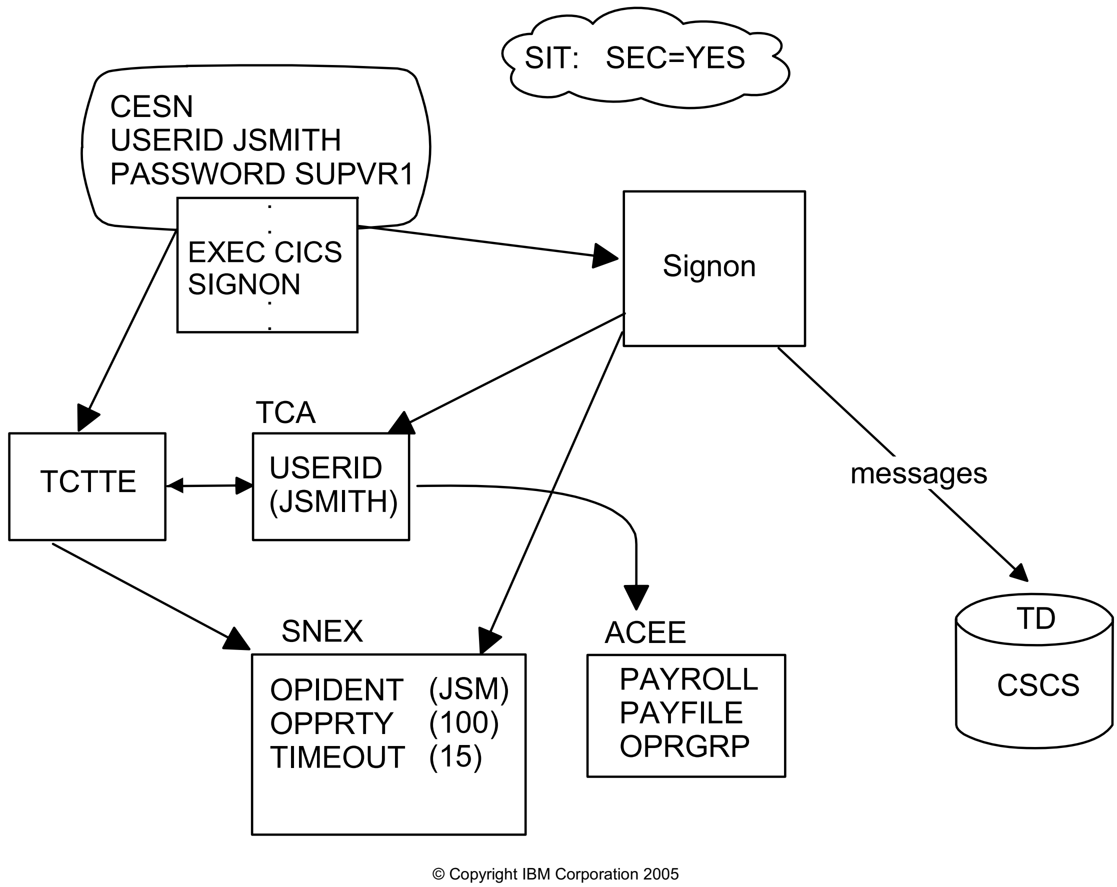


Figure 13-8. Signon CI207.0

Notes:

- User profiles are created by one of the following means:
  - CESN
  - EXEC CICS SIGNON
  - Preset terminal security: CEDA DEF TERMINAL(xxxx) USERID(userid)
- Signon creates control blocks or pointers to control blocks that define resources a user may access.

SNEX The signon extension of the TCTTE contains information from RACF's CICS segment.

ACEE Identifies profiles this user may access. ACEE is the RACF accessor control environment element.

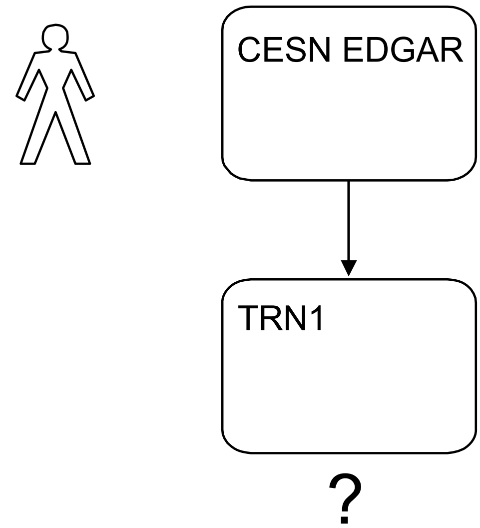
- A user is defined to RACF in the following way:

```
ADDUSER JSMITH DFLTGRP(grpname)
CICS(OPIDENT(JSM) OPPRTY(100) TIMEOUT(15)...)
```

Transaction Security

SIT:

SEC = YES
XTRAN = YES



is Edgar allowed to execute transaction TRN1?

CICS asks ESM:

"does user EDGAR have [at least] authority READ
for resource TRN1
in class TCICSTRN?"

Figure 13-9. Transaction Security CI207.0

Notes:

- `XTRAN=YES` lets CICS pass all transaction initiation requests to the ESM before starting the program assigned to this TX-Code
  - Note: The CESN transaction is an exception to this rule.
- Access READ or higher (UPDATE, CONTROL, ALTER) allows execution; access NONE (by default or explicitly) disallows execution.
- ***The region userid must be authorized to execute internal CICS system transactions; otherwise CICS will not come up.***
  - CICS system transactions are categorized for security purposes.
  - The transactions to be accessed by the Region Userid are called **category 1** transactions.
  - **SDFHSAMP(DFH$CAT1)** provides a CLIST containing RACF commands that may be executed to define the permits required by the Region Userid to RACF.

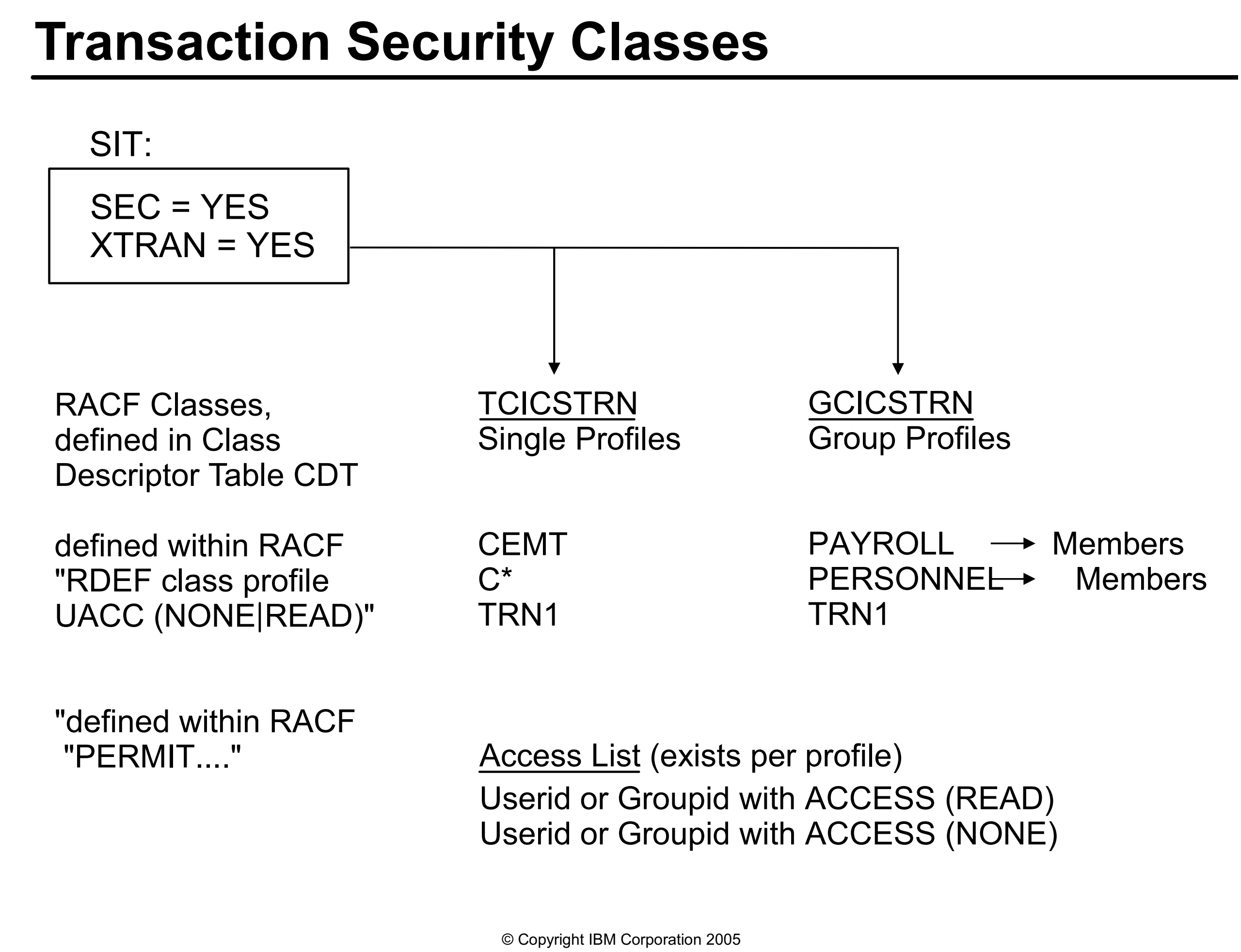


Figure 13-10. Transaction Security Classes CI207.0

Notes:

- TCICSTRN and GCICSTRN are the default class names.
- **Different resource class names may be defined if you have CICS regions with different security requirements.**
  a. `XTRAN=name`, where name is a one- to seven-character value. CICS will prefix your name with a T and G for the single and group resource class names.
  b. These class names must be defined in the RACF Class Descriptor Table (CDT). CICSnnn.SDFHSAMP(DFH$RACF) contains a job to do this.
  c. **`XTRAN=NO`** eliminates transaction security.
- Members of group profiles are single resources, in this case specific transaction codes.

Command Security

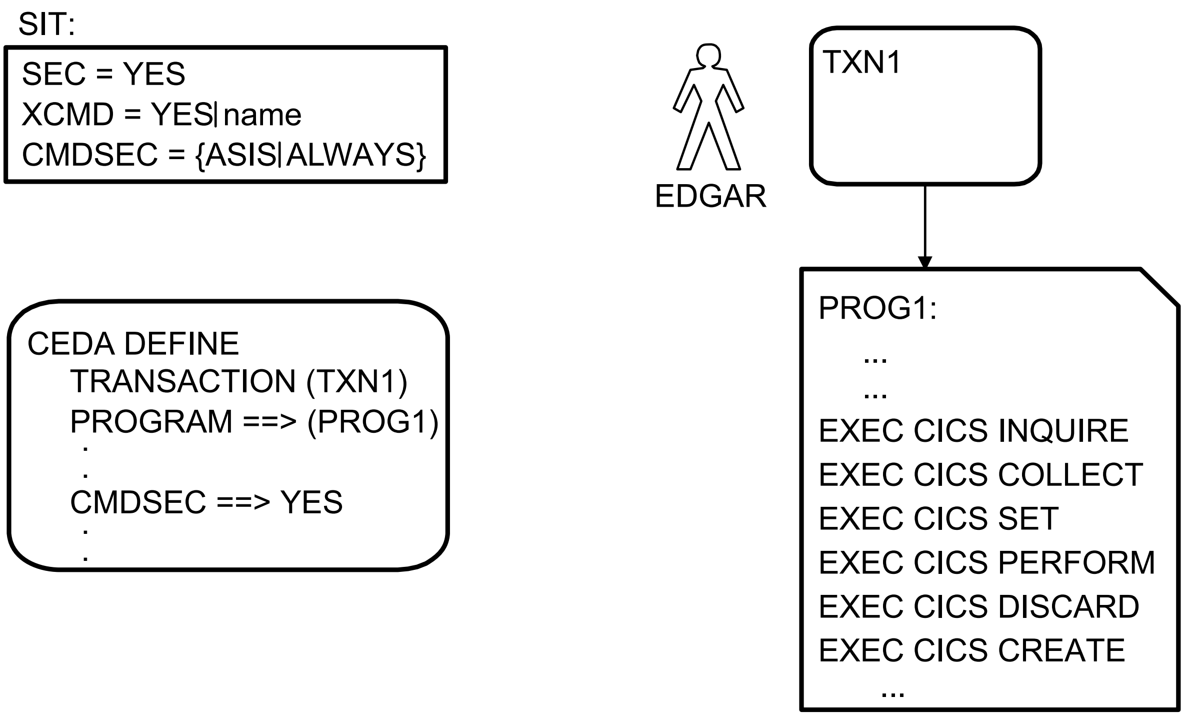


Figure 13-11. Command Security CI207.0

Notes:

- **Command security** protects the system from unauthorized use of system programming (SP) commands, including the use of CEMT actions.

INQUIRE, COLLECT	Require **READ** access
SET, PERFORM, DISCARD, CREATE	Require a minimum of **UPDATE** access to the resource type involved

- A `NOTAUTH` exceptional condition is issued to the application program if access is denied.
- Command security may be activated or deactivated on the transaction level if SIT specifies `CMDSEC=ASIS`.
- With `CMDSEC=ALWAYS`, any SPI commands are checked, regardless of the CMDSEC specification within a transaction definition.

Command Security Classes and Profiles

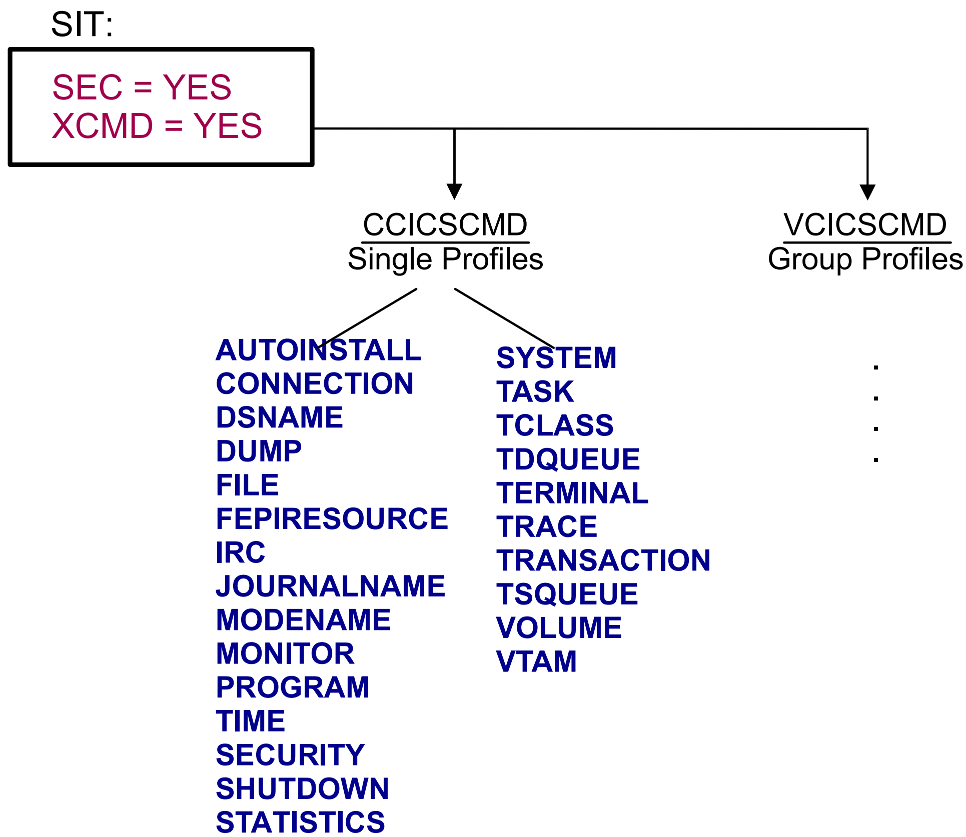


Figure 13-12. Command Security Classes and Profiles CI207.0

Notes:

- For each CICS resource type, a certain resource identifier is assigned that is used as the protected resource name within authorization requests.
  - Command security controls usage of SPI commands in combination with the CICS resource types, without regarding and distinguishing the names of the resources involved.

Resource Security

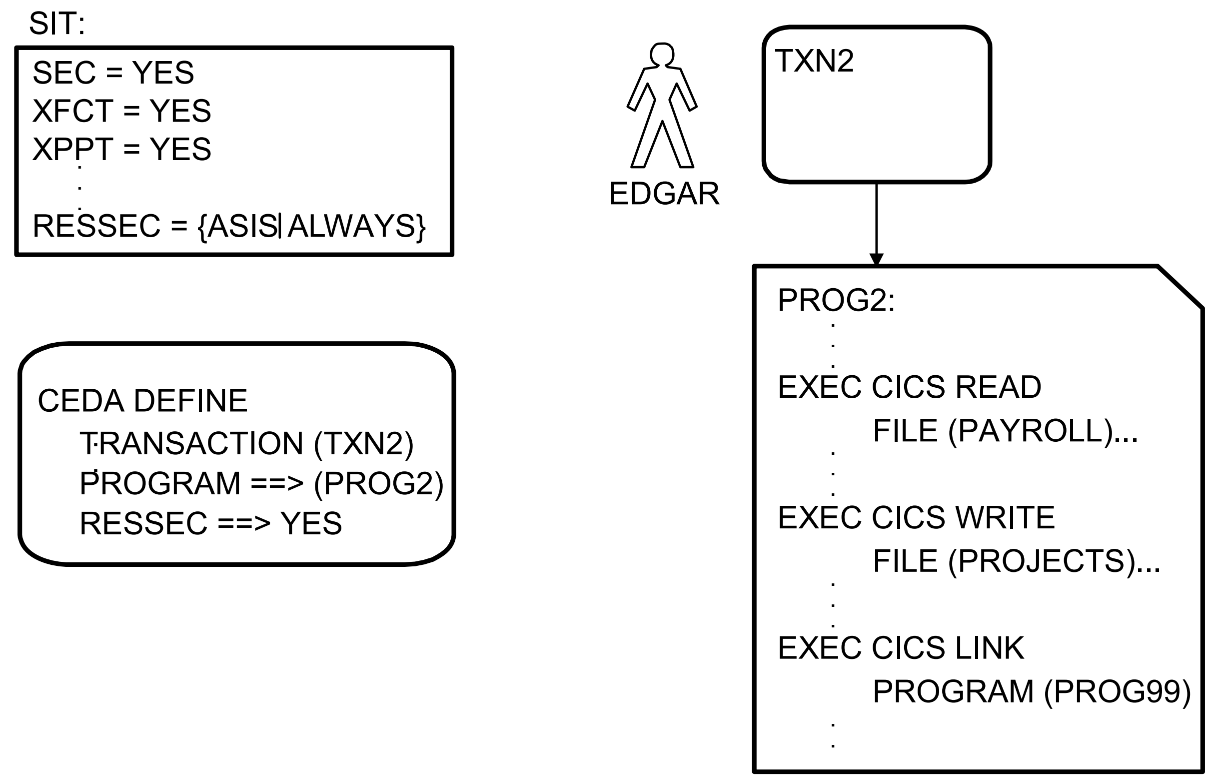


Figure 13-13. Resource Security CI207.0

Notes:

- **Resource security** provides protection of resources such as files, queues, programs, and so on, that are accessed by application programs.
- Resource security may be activated or deactivated on the transaction level if SIT specifies `RESSEC=ASIS`.
- With `RESSEC=ALWAYS`, any resource accesses will be checked, regardless of the RESSEC specification within the transaction definition.
- For accesses denied, the command issuing program is returned the `NOTAUTH` exception condition.

Resource Security Classes and Profiles

SIT:	Default Class Names		named Classes	
	Single	Group	Single	Group
SEC = YES				
XDCT = YES|name	DCICSDCT	ECICSDCT	Dname	Ename
XFCT = YES|name	FCICSFCT	HCICSFCT	Fname	Hname
XPCT = YES|name	ACICSPCT	BCICSPCT	Aname	Bname
XPPT = YES|name	MCICSPPT	NCICSPPT	Mname	Nname
XPSB = YES|name	PCICSPSB	QCICSPSB	Pname	Qname
XJCT = YES|name	JCICSJCT	KCICSJCT	Jname	Kname

Figure 13-14. Resource Security Classes and Profiles CI207.0

Notes:

- Resource Security classes may be activated and named independently of each other at the **system level**.
- At the **transaction level**, there is only the choice to use all or none of the active resource security classes.
- As for all other RACLISTed classes, changed security specifications for CICS resources, commands, and transactions may be activated without bringing CICS down or requiring a new user signon, by means of the RACF (TSO) command:

  SETROPTS RACLIST(*classname*) REFRESH

CICS Transactions

RDO group	TX-Code	RESSEC	CMDSEC
DFHOPER	CEMT	YES	YES
	CEST	YES	YES
DFHEDF	CEBR	YES	NO
	CEDF	YES	YES
DFHINTER	CECI	YES	YES
DFHSPI	CEDA	NO	NO

Figure 13-15. CICS Transactions CI207.0

Notes:

- This chart shows some of the security definitions supplied by IBM.
- Check security specifications for all CICS transactions to see if they meet your installation's specifications if you activate command security, resource security, or both.

Security Prefixing

SIT:

```
SEC = YES
XTRAN = YES
XCMD = YES
XFCT = YES
SECPRFX = YES
```

Job Control

```
// CICSP JOB 'ACCNT', USER = CIPROD
// CICS EXEC PGM = DFHSIP
.
.
.
```

Transaction Sec.	CEMT → CIPROD. CEMT	
	TX01 → CIPROD. TX01	
Command Sec.	SHUTDOWN → CIPROD. SHUTDOWN	
Resource Sec.	FILEA → CIPROD. FILEA	

Figure 13-16. Security Prefixing CI207.0

Notes:

- With SECPRFX=YES (SIT), CICS will prefix **every** resource name with its Region Userid when requesting authorization checks from an ESM.
- On the ESM side, profiles have to be defined accordingly.
- A means to define distinct authorizations for different CICS regions using only one (for example, the default) class per resource type.

Non-Terminal Security

PLT programs at startup	SIT:	PLTPISEC = {NONE|CMDSEC|RESSEC|ALL} PLTPIUSR = userid
EXEC CICS STARTed transactions	API:	E.C. START TRANSID (TRNX) USERID (userid)
TD Triggered file-oriented transactions	TD Queue Resource Definitions:	see below

```
DEFINE TDQ(ABCD)
    Type        ==> Intra
    TRAnsid     ==> TRN2
    Atifacility ==> File
    Userid      ==> ATIUSER
```


Figure 13-17. Non-Terminal Security CI207.0

Notes:

- If a userid is not specified, the following defaults become effective:

  For PLTPI programs: The Region userid

  For STARTed transactions: The userid under which the task runs that issues the `E.C.START` command

  For file-oriented ATI transactions:

  The userid specified with the `USER` parameter of the appropriate Transient Data Queue (maybe specified in the `DFHDCT TYPE=INITIAL` statement).
  If not there, the `DFLTUSER` specified in the SIT.

- Prior to CICS for MVS/ESA Version 4, non-terminal tasks could not be protected, because the userid was based on the terminal, and without a terminal, there was no userid associated to the task.

SURROGAT User Checking

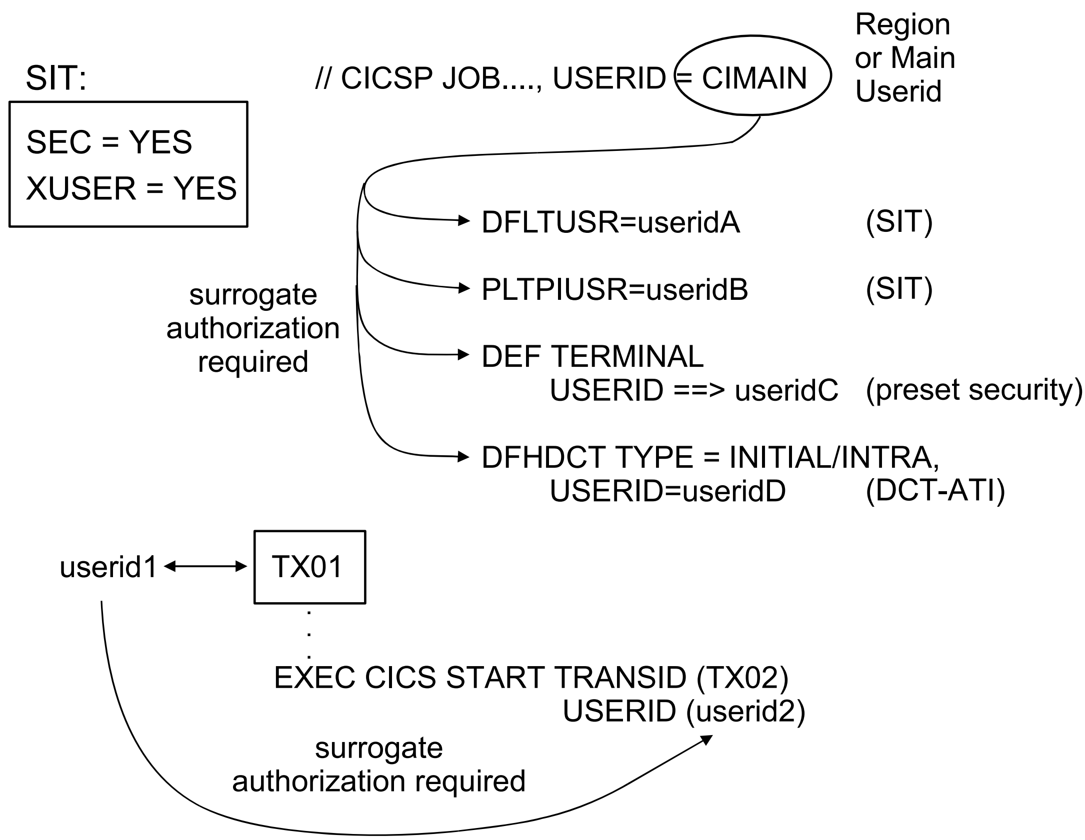


Figure 13-18. SURROGAT User Checking CI207.0

Notes:

- Protect userids from being arbitrarily used by other users.
- The Region Userid must be permitted READ access for profile `USERIDx.DFHINSTL` in RACF class SURROGAT.
- For STARTing transactions with another userid, the issuing user (userid1) needs READ for profile `USERID2.DFHSTART` in class SURROGAT.
- These checks are performed only if **`XUSER=YES`** is coded in the SIT.

Securing External Requests

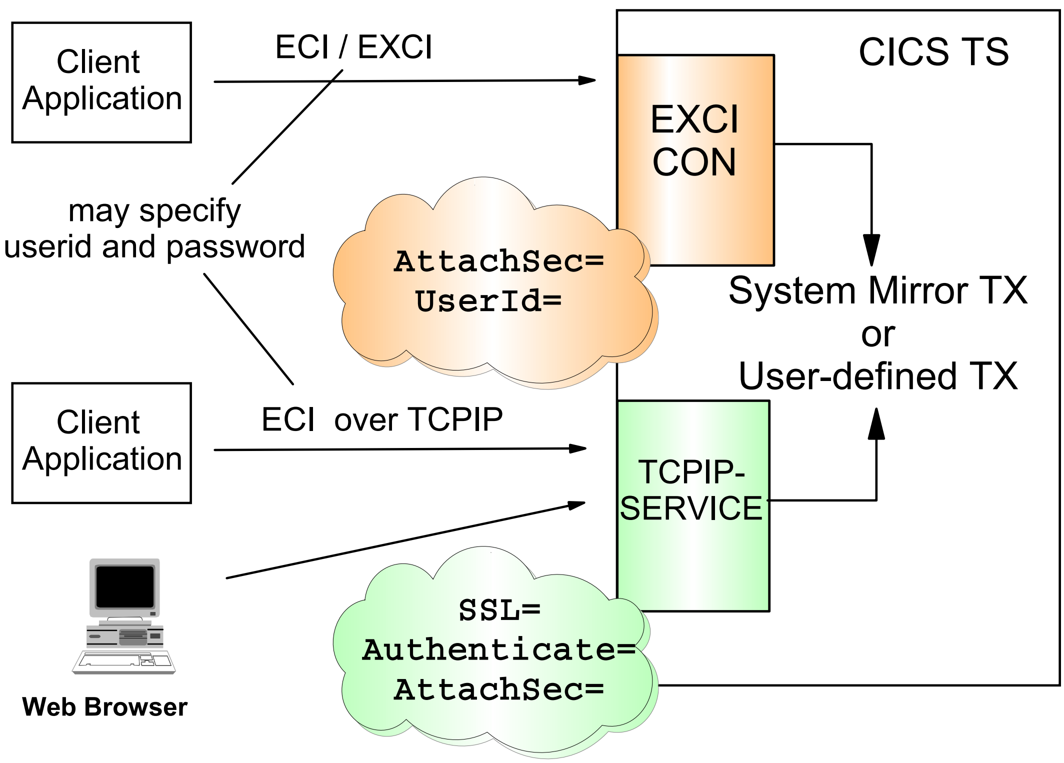


Figure 13-19. Securing External Requests CI207.0

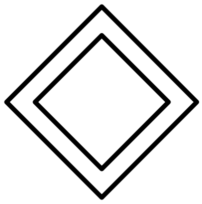
Notes:

In all cases, where CICS service is invoked by applications and systems that are connected over the network or by some other way to the CICS region, the CICS resources that control the appropriate connections provide security-related parameters that allow security requirements to be specified.

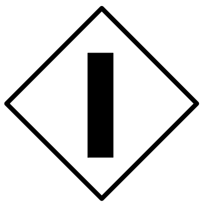
- For local ECI and EXCI from a local CTG or batch client, it is the CONNECTION definition that lets you determine, for example, whether or not incoming requests have to provide a userid and a password. An alternative could be to predefine a userid that is associated to all requests from that connection.
- For TCP/IP network connections, it is the TCPIPSERVICE resource that offers the security options, including the use of SSL, which means that a RACF userid is mapped to a Digital Certificate that has to be provided by the client.

The result is always that a particular RACF userid (maybe two in some cases) is or are associated to the transaction(s) to be run on behalf of the client requests, and from there on, security checking is the same as for terminal-initiated transactions.

Recommendations



Consider transaction security
to be a MUST for production systems



Command security is recommended,
at least to allow common
use of CEMT INQUIRE functions

Handle resource security with care;
apply only if you have real requirements

Figure 13-20. Recommendations CI207.0

Notes:

Remember, applying security reasonably means protecting resources and services as much as *necessary*, not more!

CICS Signoff

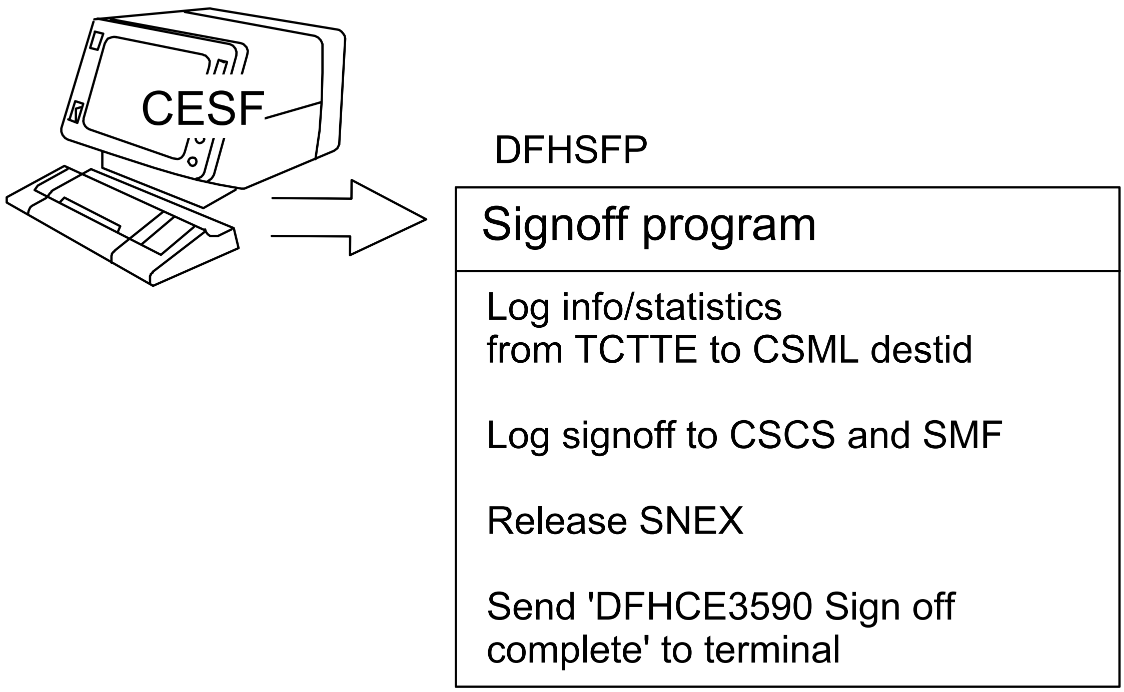


Figure 13-21. CICS Signoff CI207.0

Notes:

- Users should develop the practice of signing off before leaving their terminal unattended.
- This removes security privileges by releasing the operator's SNEX (signon extension control block) and redirecting pointers to the control blocks representing the `DFLTUSER` userid.
- Signoff may be accomplished by a program issuing `EXEC CICS SIGNOFF`.

Terminal Timeout

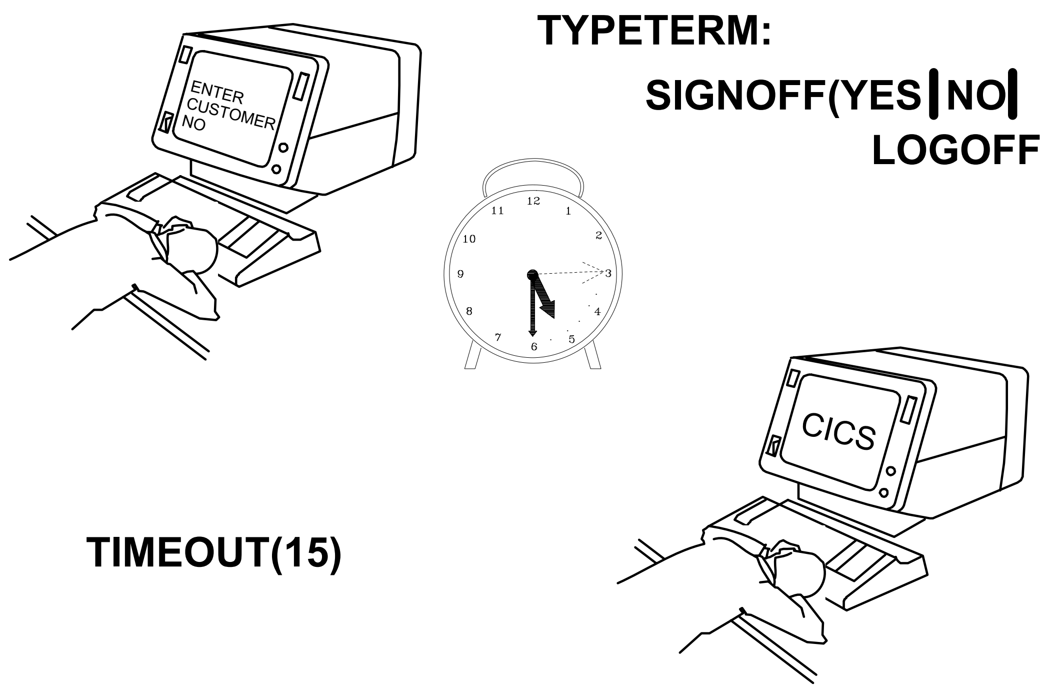


Figure 13-22. Terminal Timeout CI207.0

Notes:

- **TIMEOUT**, in the CICS segment of the RACF database, forces an unused terminal to time out after the specified number of minutes elapse.
- After this period, the SIGNOFF option in the TYPETERM is taken.

Security (Summary)

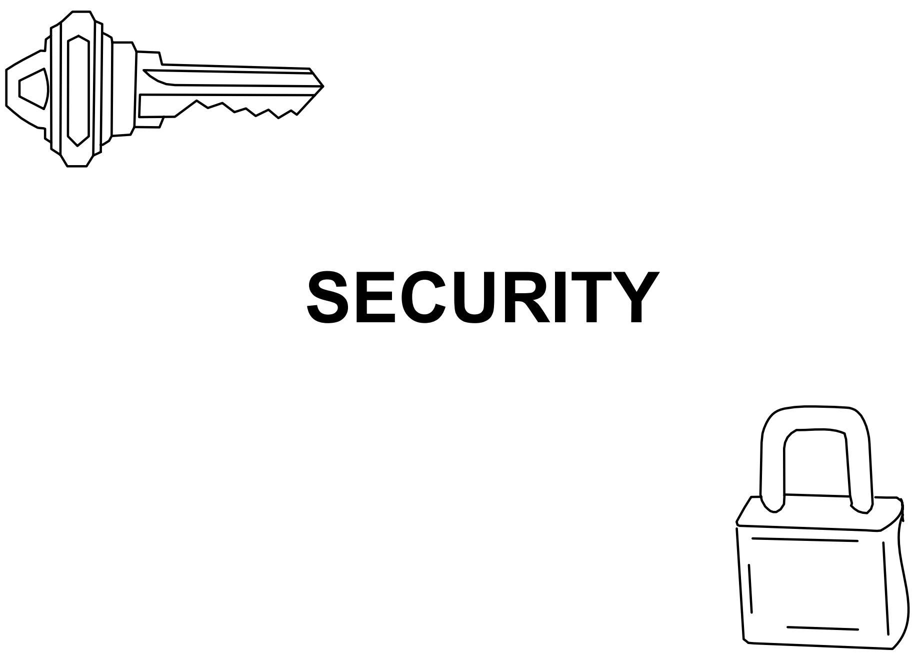


Figure 13-23. Security (Summary) CI207.0

Notes:

- A couple of security definitions are required to bring up a CICS TS region regularly.
- Based on the CICS region userid, CICS's access to other system components may be protected, thus determining:
  - Which CICS may use which VTAM ACB name
  - Which CICS may request SMSVSAM service for VSAM RLS
  - Which CICS may read or update which Coupling Facility structures for log streams and shared temporary storage pools
- Security-related definitions may be made for CICS users accessing CICS TS resources, and for CICS as a system that uses other system services and resources.
- The SIT defines whether or not CICS resources are secured, and what types of resources are being protected.
- Users must sign on to implement their security profile, otherwise the profile of a *Default User* is checked for their actions.

- RACF resources can be either single or group definitions. While single is more granular, grouping is the preferred way to protect resources.
- CICS supports three kinds of security:
  - Transaction security protecting transaction codes
  - Command security protecting the use of SPI commands
  - Resource security protecting single (application) resources.

Unit Summary

Having completed this unit, you should be able to:

- Explain CICS security implementation to the external security manager (ESM) administrator
- State what security definitions are required to run CICS TS under z/OS, and to allow CICS TS to request service from other system components such as VTAM, VSAM, and so on
- Explain and contrast the three basic types of security that may be used
- Define the types of resources that may be protected
- Explain how multiple CICS TS regions accessing the same ESM can be protected independently from each other
- State the means and parameters that support protection of non-terminal-oriented tasks

Figure 13-24. Unit Summary CI207.0

Notes:

Unit 14. Basic Problem Determination

What This Unit Is About

CICS includes several sources of information to help you perform problem determination. This unit introduces you to what each of them provides, and shows you where to go for more help.

What You Should Be Able to Do

After completing this unit, you should be able to:

- Outline a simple approach to problem determination
- List the major sources of debugging information
- Select the proper method of handling CICS messages
- List the various types of traces available in CICS
- State CICS specifications for controlling and printing traces
- State CICS specifications for controlling and processing dumps

References

CICS Problem Determination Guide, SC34-6441

CICS Operations and Utilities Guide, SC34-6431

CICS Diagnosis Reference, LY33-6110

Unit Objectives

After completing this unit, you should be able to:

- Outline a simple approach to problem determination
- List the major sources of debugging information
- Select the proper method of handling CICS messages
- List the various types of traces available in CICS
- State CICS specifications for controlling and printing traces
- State CICS specifications for controlling and processing dumps

Figure 14-1. Unit Objectives CI207.0

Notes:

14.1 General Considerations

Problem Determination Approach

- **Planning**
  - Track changes
  - Understand information sources
  - Educate the users
- **When problems are nontrivial**
  - Ask questions
  - Collect information
  - Classify the symptom(s)
  - Determine the cause(s)

Figure 14-2. Problem Determination Approach CI207.0

Notes:

- The best way to solve problems is to anticipate them. This shortens the discovery cycle and can even prevent problems.
- The most important planning activity is setting up a system that allows you to determine the nature and timing of changes, because many problems are caused by a failed change to the system.
- Once you encounter a problem, classify the symptom in a way that will help the IBM support center match your symptom with the description of a problem already discovered by another user.
- The *CICS Operations and Utilities Guide, SC34-6431*, has sample incident reports in the “Operating Procedures” chapter which you can tailor to your own requirements for problem tracking.

Major Sources of Information

- CICS Messages
- CICS Transactions
- Dump Facility
- Trace Facility

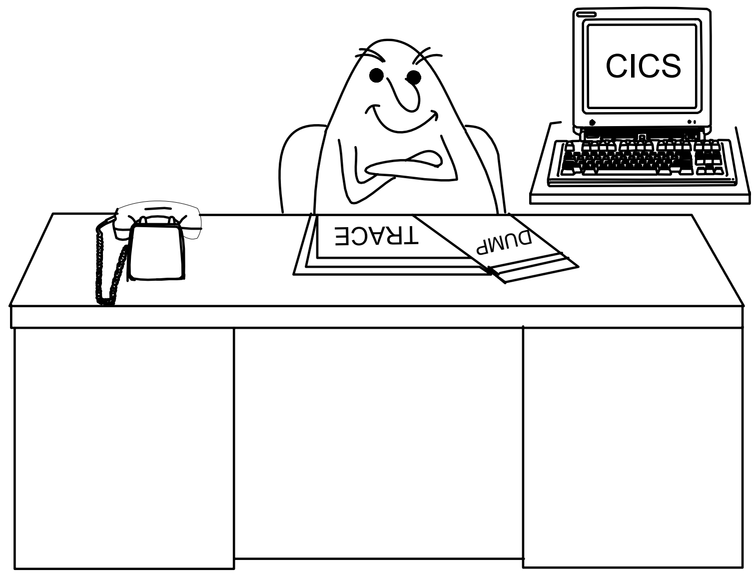


Figure 14-3. Major Sources of Information CI207.0

Notes:

These sources provide all the data normally needed for problem determination. The remainder of this unit explains what has to be done to collect and use the information.

CICS TS for z/OS, Version 2.3 provides an improved level of support for the use of workstation-based and host-based debuggers, particularly to support the IBM Debug Tool.

- We regard this an application programming issue that is not covered by this course. For details, refer to the “Improved Interactive Debugging” section of the *CICS Transaction Server for z/OS Version 3.1 Release Guide, GC34-6421*.

CICS-Supplied Transactions

- CEMT (Master Terminal)
  - Inquire on Resources
- CEDF, CEDX (Execution Diagnostic)
  - Test applications interactively
- CMAC (Display Messages and Codes)
  - Online Message Explanations
- CSFE (Field Engineer Support)
  - Diagnosing Terminals and Systems
- CEDA (Resource Definition Online)
  - Define and Review Resources
- CEBR (Browse Temporary Storage)
  - Browse TD Intra or TS Queue contents
- CECI, CECS (Command Interpreter)
  - Execute Commands

Figure 14-4. CICS-Supplied Transactions CI207.0

Notes:

- Use these transactions before attempting more detailed analysis with dumps and traces.
  CICS Supplied Transactions, SC34-6432, contains more information.
- CEDF, CEDX, CECI, CECS, and CEBR are also useful for application developers. They are described in the *CICS Application Programming Guide, SC34-6433*.
- To view the explanation of a CICS message online, use CMAC. Define the CICS Message File, also called CMAC, as a file resource. This definition, a part of group DFHCMAC, is included along with other IBM-supplied resources when you initialize DFHCSD.
- The diagnostic transaction CSFE should be used only by experienced system programmers and IBM service personnel. Some of its functions impact system performance.
- Another transaction, CETR, controls trace, and is covered later in this unit.

CICS Message Format

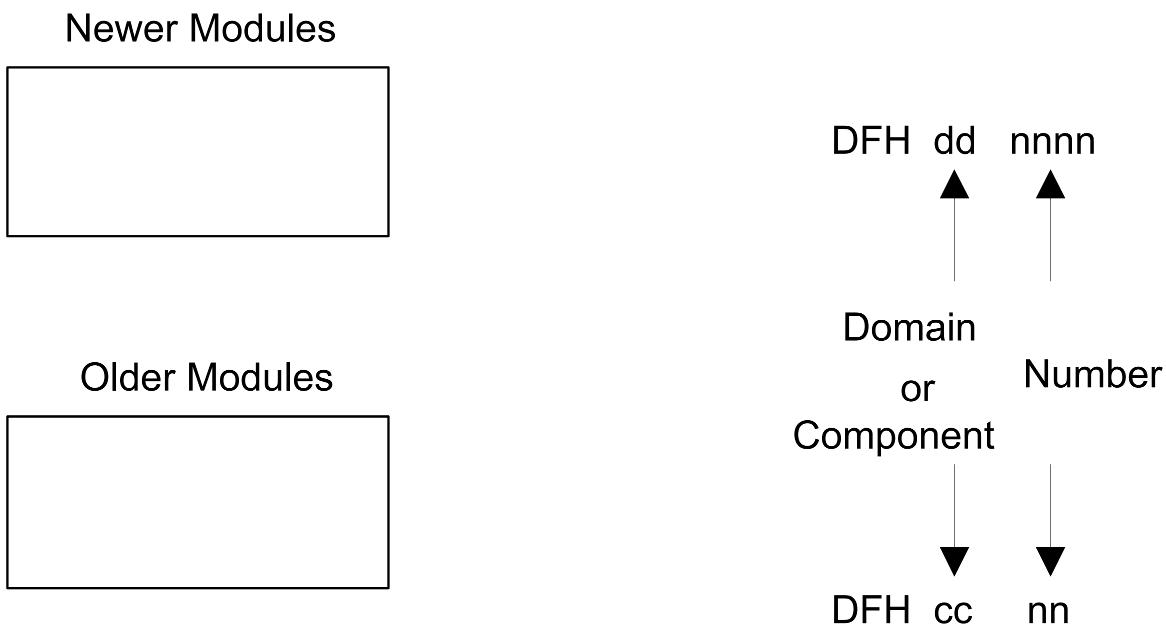


Figure 14-5. CICS Message Format CI207.0

Notes:

- All CICS messages consist of:
  - The prefix **DFH**
  - Two characters that tell which domain or component issued the message
    - Modules rewritten for CICS/ESA or CICS TS, respectively, use alphabetic characters.
    - For older modules, two digits identify the management program.
  - A sequential number
- *CICS Messages and Codes, GC34-6442*, lists the messages in sequence, along with possible causes and suggested actions.

Message Disposition

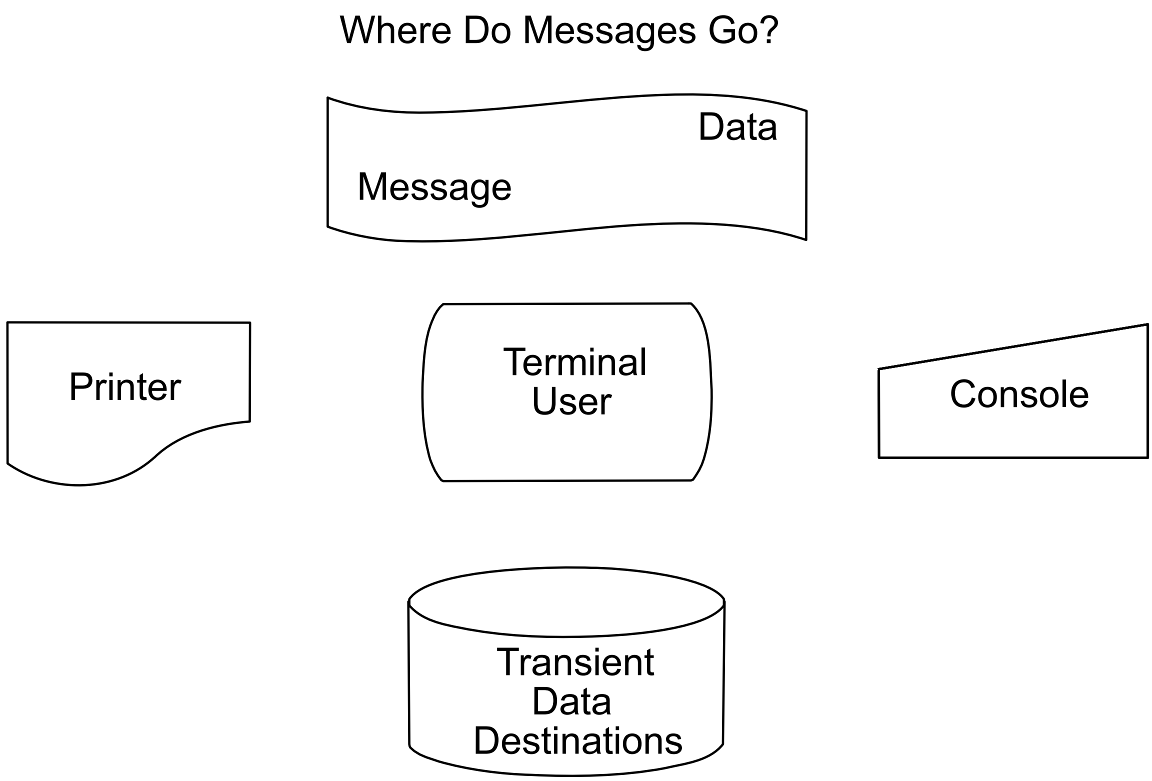


Figure 14-6. Message Disposition CI207.0

Notes:

When a problem occurs, the next course of action should be to examine all available messages.

- CICS routes urgent messages of system-wide interest to an operator console. If you define CICS as a subsystem to z/OS, these messages include the VTAM APPLID to help determine which CICS region is affected.
- You can exercise some control over the format and route codes for these messages. See "ConsolE Message Formatting" in the *CICS Transaction Server for z/OS Version 3.1 Installation Guide, GC34-6426*, for details.
- The terminal user receives transaction abend messages.
- CICS sends most other messages to transient data destinations.
- Check SYSOUT data sets for messages from CICS batch utilities and error messages when CICS fails to initialize.

Controlling Messages

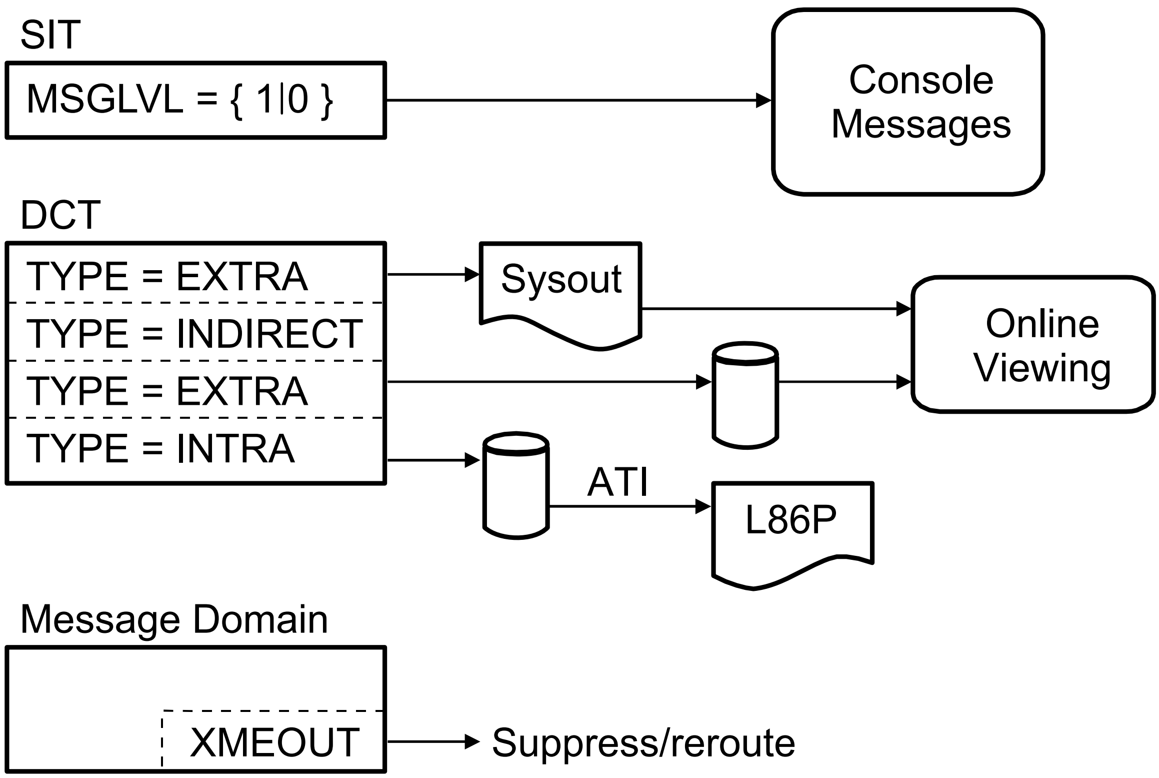


Figure 14-7. Controlling Messages CI207.0

Notes:

CICS allows control of what messages are produced and how they are viewed.

- If `MSGLVL=0` is selected, only critical messages are displayed.
- There are several choices for viewing messages routed to transient data:
  - To send informational messages to SYSOUT and use SDSF to see them.
  - To direct messages to a sequential file and BROWSE it with ISPF/PDF.
  - Messages which require operator monitoring, such as transaction and terminal errors, could be printed on a CICS printer using automatic task initiation.
- Use the XMEOUT exit of the Message Domain to suppress or reroute messages before CICS writes them.

Transient Data Destination

TDqueue:	Message types	Recommendation TYPE=	Recommendation Disposition
CSMT CSTL CSNE	Transaction abends Terminal errors	INTRA	USE DFH$TDWT to print on-line
CSCS CSML	Sign on/off Security violation	INDIRECT	Indirectname=CSSL SYSOUT Sample RDO group DFHDCTG
CSDL CADL CRDI	RDO changes, resource installation		
CDUL	Transaction dump info		
CSFL	Dynamic allocation		
CAIL	Terminal Autoinstall log		

Figure 14-8. Transient Data Destination CI207.0

Notes:

- This chart summarizes the most important CICS message destinations and recommended ways to deal with them. All CICS destinations are defined in the *CICS Resource Definition Guide, SC34-6430*.
- If SDSF is not available, define CSSL as a sequential disk file, and use ISPF/PDF to inspect the messages.
- CSDL could be captured on a separate file and used to reconstruct DFHCSD updates in case regular backup fails.
- For critical messages, define an intrapartition destination which triggers a transaction to print this information on a terminal printer. There is a CICS-provided sample program DFH$TDWT that may be used for this purpose, probably with some enhancements to meet special requirements.

14.2 CICS TS Trace Facility

Types of Traces

- Normal CICS system
- CICS exception
- CICS VTAM exits
- Program checks and abends
- VTAM buffer

Figure 14-9. Types of Traces CI207.0

Notes:

This chart summarizes the types of tracing CICS supports.

Normal	User program and management module requests
Exception	First failure data capture by CICS domains
VTAM exit	CICS Terminal Management's use of VTAM exits
PC/Abend	Program checks and MVS abend conditions in a system dump
VTAM buffer	Supplied by VTAM, written to GTF, and often useful for correlating terminal traffic with CICS events

CICS always invokes exception and PC/abend tracing as needed.

CICS Trace Overview

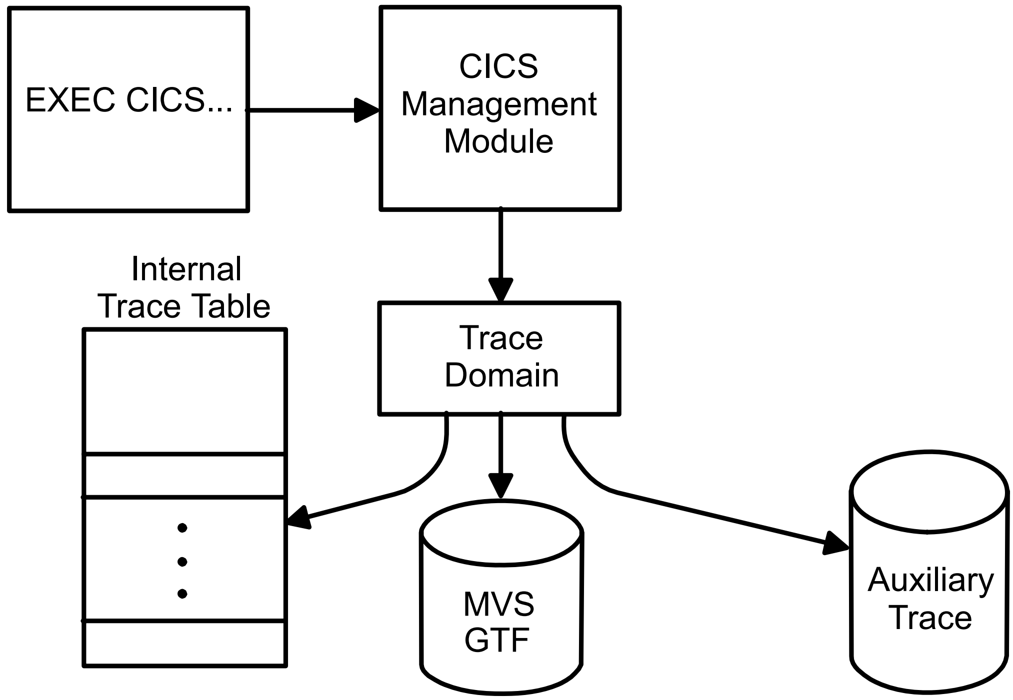


Figure 14-10. CICS Trace Overview CI207.0

Notes:

- CICS trace provides a record or trail of transaction and system activity.
- Trace points are included at specific points in CICS code. Some trace points are used to make exception traces when exception conditions occur, and some are used to trace the mainline execution of CICS code.

  Trace points of the latter type each have a “level” attribute associated, the value of which may vary, in principle in the range 1-32. The higher this value, the more detailed the information provided by the trace entries. In practice, nearly all mainline trace points have a trace level of 1 or 2.
- Trace activity can be directed to any combination of:
  - An in-storage wraparound table
  - The Generalized Trace Facility (GTF) of MVS
  - An auxiliary data set managed by CICS.

Selecting Trace Options

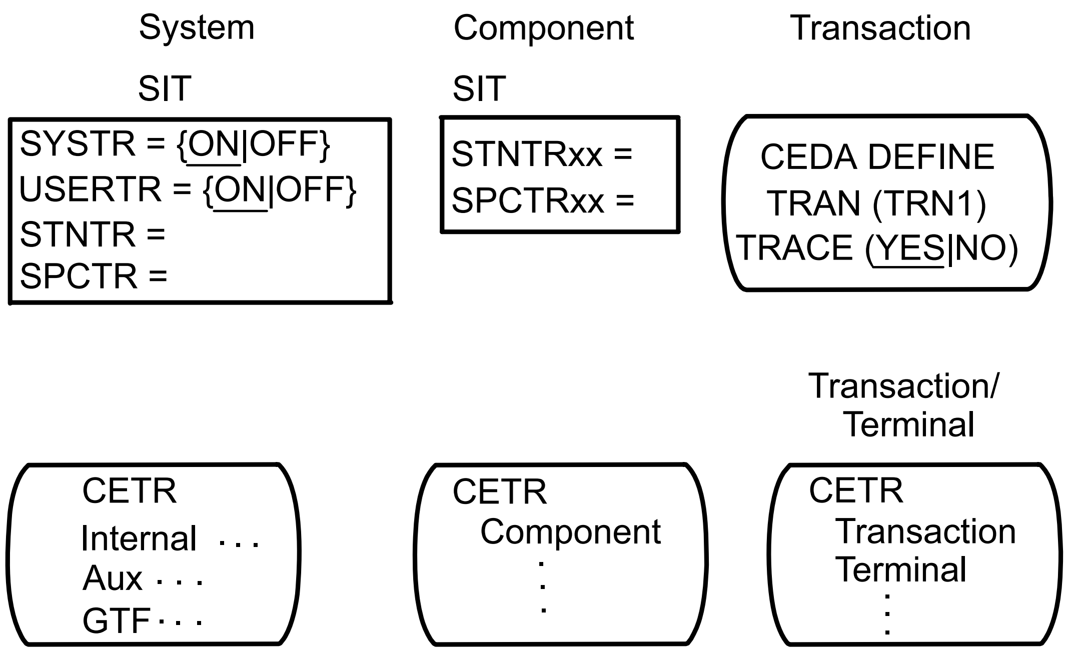


Figure 14-11. Selecting Trace Options CI207.0

Notes:

- At the **system** level, the following options are available:
  - Set the master trace flag **SYSTR=ON** to activate trace points in CICS modules.
  - Allow user programs to issue ENTER TRACE commands via **USERTR=ON**.
  - Select the level of standard or special tracing with **STNTR** and **SPCTR**.
- Use CETR or SIT to control the level of tracing by **component**.
- Trace for a certain task can be enabled or disabled by the transaction definition.
- CETR allows for specifying standard or special tracing, or suppressing of tracing altogether, for a particular transaction at a particular terminal, or for all transactions at a terminal.
- Trace options may be set using the system programming interface:
  - EXEC CICS SET TRACEFLAG sets global trace flags.
  - EXEC CICS SET TRACETYPE selects standard or special trace by component.

Trace Control Transaction CETR

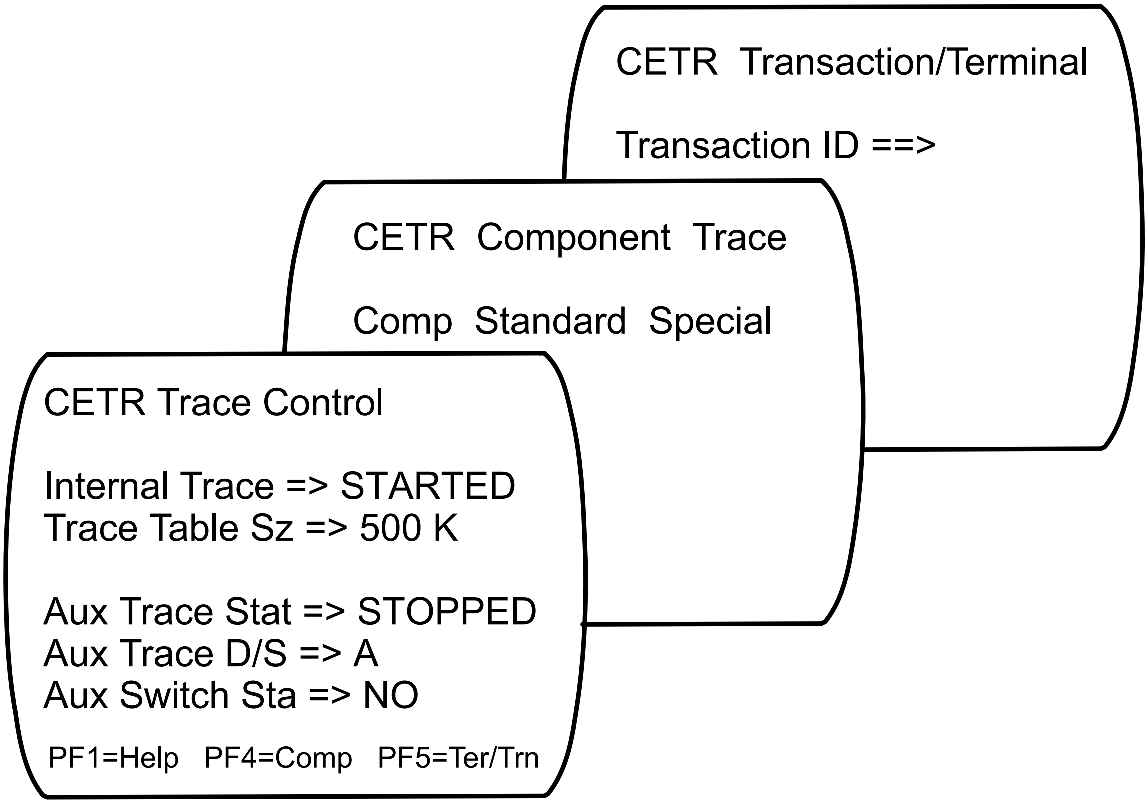


Figure 14-12. Trace Control Transaction CETR CI207.0

Notes:

CETR allows for selection of what types of traces to record, at three levels of detail:

- At the **system** level, specify whether trace is to be used, the size of the internal trace table, and where output is directed.
- For each **component**, select standard or special tracing.
- Trace options may be selected by **terminal**, **transaction**, or both. This capability allows for tracing a certain transaction only when it is running at a particular terminal, for example.

Trace Destinations

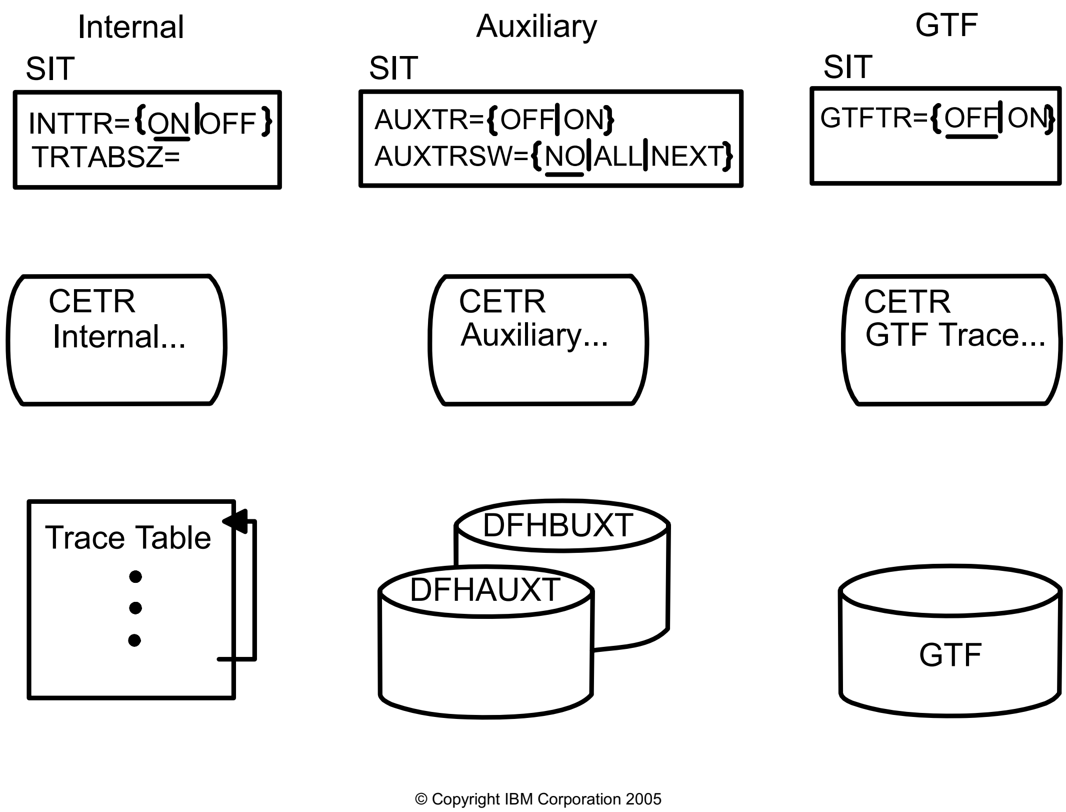


Figure 14-13. Trace Destinations CI207.0

Notes:

CICS supports tracing to any of three destinations, using the SIT and the CETR transaction.

- For internal trace, specify the table size, which defaults to 16K bytes. When the table fills, recording “wraps around” to the beginning and continues.
- Auxiliary tracing is directed to one or two sequential data sets. Automatic switching of data sets can be requested when one extent fills.
- GTF tracing may be helpful in diagnosing system problems, since CICS information is combined with traces from other components, such as VTAM.
- `EXEC CICS SET TRACEDEST` can be used to control trace recording programmatically. See the *CICS System Programming Reference, SC34-6435*, for details.

Formatting and Printing Traces

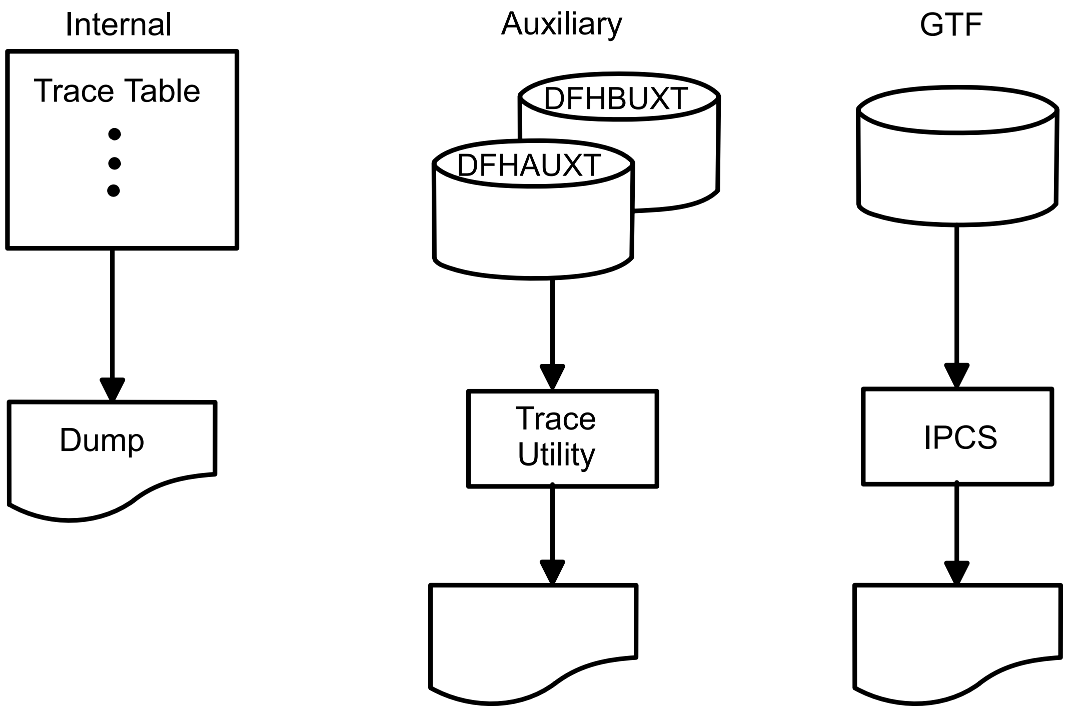


Figure 14-14. Formatting and Printing Traces CI207.0

Notes:

- CICS includes the internal trace table in transaction and system dumps.
- The **Trace Utility Program**, `DFHTU640` (for CICS TS 3.1) or `DFHTU530` (for CICS TS 1.3) respectively, prints auxiliary trace files. The depth of information to be printed for each trace entry can be controlled by the following options:

  ABBREV Abbreviated, one-line-per-entry summary form of trace print.

  SHORT As abbreviated, plus some additional helpful elements.

  FULL Formats all the data from each entry.

  Independent from the depth option, printing may be requested by transaction ID, time range, terminal, specific task, and trace points. Usage of `DFHTU*` is documented in the *CICS Operations and Utilities Guide, SC34-6431*.
- Use IPCS to print CICS GTF trace entries. Similar selections as in the Trace Utility may be used. The trace formatting routine named `DFHTR640/*530` (contained in the SDFHLINK library) must be made available to IPCS in order to use this option.

ABBREViated Trace Output

```
CICS - AUXILIARY TRACE FROM 09/06/02 - APPLID CICSTST
 SELECTIVE TRACE PRINT PARAMETERS:
 ABBREV
 TIME OF FIRST ENTRY ON THIS PAGE 10:33:57.9956673439             PAGE 00005
00107 QR    AP 00E1 EIP    ENTRY WRITE  0004,00140478 ....,08000604 ....          =000171=
00107 QR    AP 04E0 FCFR   ENTRY WRITE
                   FILEA,00000000,0014070B,80140449,50,NO,NO,NO,KEY,NO,NO       =000172=
00107 QR    AP EA00 TMP    ENTRY LOCATE                FCT,FILEA                  =000173=
00107 QR    AP EA01 TMP    EXIT  LOCATE                FCT,FILEA,0EBFC3D8,NORMAL  =000174=
00107 QR    NQ 0301 NQED   ENTRY ENQUEUE               0E24A200,0EBDC50C ,
                               00000004,0EBDC510 , 00000006,NO,.{..007822       =000175=
00107 QR    SM 0301 SMGF   ENTRY GETMAIN               15A99AAC , 00000052,
                               9C,NO,00,NQEA                                    =000176=
00107 QR    SM 0302 SMGF   EXIT  GETMAIN/OK            0E24B300                   =000177=
00107 QR    NQ 0302 NQED   EXIT  ENQUEUE/OK                                       =000178=
....
00107 QR    DS 0004 DSSR   ENTRY WAIT_MVS       FCIOWAIT,0EBDC460,NO,IO,FILEA     =000185=
TCP   QR    DS 0005 DSSR   EXIT  WAIT_OLDW/OK                                     =000186=
TCP   QR    XM 0801 XMSR   ENTRY INQUIRE_MXT                                      =000187=
TCP   QR    XM 0802 XMSR   EXIT  INQUIRE_MXT/OK        14,2                       =000188=
TCP   QR    DS 0004 DSSR   ENTRY WAIT_OLDW
                               TCP_NORM,00057DD0,CSTP,NO,IDLE,DFHZDSP           =000189=
00107 QR    DS 0005 DSSR   EXIT  WAIT_MVS/OK                                      =000190=
00107 QR    AP 04E1 FCFR   EXIT  WRITE/OK          00000000,LENGTH_OK,YES,NO      =000191=
00107 QR    AP 00E1 EIP    EXIT  WRITE                 OK
                                      00F4,00000000 ....,00000604 ....          =000192=
00107 QR    AP 00E1 EIP    ENTRY SEND-MAP 004,00140478 ....,08001804 ....         =000193=
....
00107 QR    PG 0601 PGLD   ENTRY LOAD    DFH$AGA,TASK_LIFE,MAPSET,NO              =000202=
00107 QR    DD 0301 DDLO   ENTRY LOCATE  0E227B70,0EB83978,PPT,DFH$AGA            =000203=
00107 QR    DD 0302 DDLO   EXIT  LOCATE/OK    D7D7E3C5 , 0EC4B8C8                 =000204=
```


Figure 14-15. ABBREViated Trace Output CI207.0

Notes:

- When formatted with the **ABBREV** option, the trace data captured is reduced to a one-line summary per trace entry.
  Note that some lines in the visual had to be split to fit to the page width.
- The display shows, from left to right:
  a. Task ID
  b. TCB used (QR names the quasi-reentrant TCB)
  c. Trace ID, made up of the domain or component ID and a four-digit number
  d. The domain gate called
  e. The function called
- The entries shown cover the event of a WRITE I/O to the FILEA file when using the AADD transaction to add an entry to FILEA.
  - Because no other task was active at the time the trace was taken, our AADD task with task ID 00107 regained control immediately after the file I/O completed.

SHORT Trace Output

```
CICS - AUXILIARY TRACE FROM 09/06/02 - APPLID CICSTST
 SELECTIVE TRACE PRINT PARAMETERS:
 SHORT
00107 QR    AP 00E1 EIP   ENTRY WRITE REQ(0004) FIELD-A(00140478 ....)
                                FIELD-B(08000604 ....)
                                RET-500C305C 10:33:57.9957401486 00.0000011328 =000171=
00107 QR    AP 04E0 FCFR  ENTRY WRITE FILE_NAME(FILEA)
                                ENVIRONMENT_IDENTIFIER(00000000)
                                RECORD_ADDRESS(0014070B)
                                RECORD_ID_ADDRESS(80140449) RECORD_LENGTH(50)
                                BACKOUT(NO) MASS_INSERT(NO) PRIVILEGED_REQUEST(NO)
                                RECORD_ID_TYPE(KEY) CONDITIONAL(NO)
                                BYPASS_SECURITY_CHECK(NO)
                                RET-8E557622 10:33:57.9957463439 00.0000061953 =000172=
00107 QR    AP EA00 TMP   ENTRY LOCATE TABLE(FCT ) KEY(FILEA    )
                                RET-8E8FBA6A 10:33:57.9957554455 00.0000091015 =000173=
00107 QR    AP EA01 TMP   EXIT  LOCATE TABLE(FCT ) KEY(FILEA    )
                                ENTRY_ADDRESS(0EBFC3D8) RESPONSE(NORMAL)
                                RET-8E8FBA6A 10:33:57.9957592814 00.0000038359 =000174=
....
00107 QR    DS 0004 DSSR  ENTRY WAIT_MVS RESOURCE_TYPE(FCIOWAIT) ECB_ADDRESS(0EBDC460)
                                PURGEABLE(NO) WLM_WAIT_TYPE(IO)RESOURCE_NAME(FILEA)
                                RET-8E9C5F18 10:33:58.0024140788 00.0001227031 =000185=
....
00107 QR    DS 0005 DSSR  EXIT  WAIT_MVS/OK
                                RET-8E9C5F18 10:33:58.0111937578 00.0000057578 =000190=
00107 QR    AP 04E1 FCFR  EXIT  WRITE/OK ACCMETH_RETURN_CODE(00000000)
                                LENGTH_ERROR_CODE(LENGTH_OK) TERMINATE_STRING(YES)
                                TERMINATE_REMOTE_REQUEST(NO)
                                RET-8E557622 10:33:58.0112239140 00.0000301562 =000191=
00107 QR    AP 00E1 EIP   EXIT  WRITE OK
                                REQ(00F4) FIELD-A(00000000 ....) FIELD-B(00000604 ....)
                                RET-500C305C 10:33:58.0112333437 00.0000094296 =000192=
....
```


Figure 14-16. SHORT Trace Output CI207.0

Notes:

- These are the first three entries and the last section of the same trace output, printed with the `SHORT` option.
- Note that trace sequence numbers are the same for corresponding trace entries.
- The SHORT formatting includes time intervals between trace entries.

ALL/FULL Trace Output

```
CICS - AUXILIARY TRACE FROM 09/06/02 - APPLID CICSTST
 SELECTIVE TRACE PRINT PARAMETERS:
 FULL

AP 00E1 EIP ENTRY WRITE       REQ(0004) FIELD-A(00140478 ....) FIELD-B(08000604 ....)
                              TASK-00107 KE_NUM-0037 TCB-QR   /008D3B60 RET-500C305C
                              TIME-10:33:57.9957401486 INTERVAL-00.0000011328       =000171=
AP 04E0 FCFR ENTRY - FUNCTION(WRITE) FILE_NAME(FILEA)
                              ENVIRONMENT_IDENTIFIER(00000000) RECORD_ADDRESS(0014070B)
                              RECORD_ID_ADDRESS(80140449) RECORD_LENGTH(50) BACKOUT(NO)
                              MASS_INSERT(NO) PRIVILEGED_REQUEST(NO) RECORD_ID_TYPE(KEY)
                              CONDITIONAL(NO) BYPASS_SECURITY_CHECK(NO)
                              TASK-00107 KE_NUM-0037 TCB-QR   /008D3B60 RET-8E557622
                              TIME-10:33:57.9957463439 INTERVAL-00.0000061953        =000172=

1-0000  00880000 00000038 00000000 00000000  B409C183 BC800000 05000100 00000000  *.h...............Ac............*
  0020  00000000 C6C9D3C5 C1404040 00000000  00000000 00000000 00000000 00000000  *....FILEA   ....................*
  0040  00000000 00000000 00000000 00000000  00000000 0014070B 80140449 00000050  *..............................&*
  0060  00000000 00000000 00000000 00000000  00000200 00000000 01020200 01020002  *................................*
  0080  00000200 00100000                                                         *........                        *
....
....
NQ 0301 NQED  ENTRY - FUNCTION(ENQUEUE) POOL_TOKEN(0E24A200) ENQUEUE_NAME1(0EBDC50C ,
                              00000004) ENQUEUE_NAME2(0EBDC510 , 00000006)
                              TASK-00107 KE_NUM-0037 TCB-QR   /008D3B60 RET-8E9C24A0
                              TIME-10:33:57.9957760627 INTERVAL-00.0000167812        =000175=

1-0000  00400000 00000128 00000000 00000000  BE800000 00000000 01000100 0E24A200  *. ............................s.*
  0020  0EBDC50C 00000004 0EBDC510 00000006  00000000 02000000 00000000 00000000  *..E.......E.....................*
2-0000  0EC072A0                                                                  *.{..                            *
3-0000  F0F0F7F8 F2F2                                                             *007822                          *
....
....
```


Figure 14-17. ALL/FULL Trace Output CI207.0

Notes:

- Some selected entries of the same trace output formatted with the **FULL** or **ALL** option, which is the default.
- The complete data contained in the trace record is shown, with the extracts of storage areas listed in dump format.
- Again, trace numbers match with the ones in the ABBREV or SHORT trace output, and thus allow for quick and easy retrieval of entries and sections that are of special interest for a problem determination activity.
- The full documentation of trace entries is provided by *CICS Trace Entries, SC34-6443*. This book lists CICS trace entries alphanumerically, and contains the complete interpretation tables for each entry.

14.3 CICS TS Dump Facilities

Types of Dumps

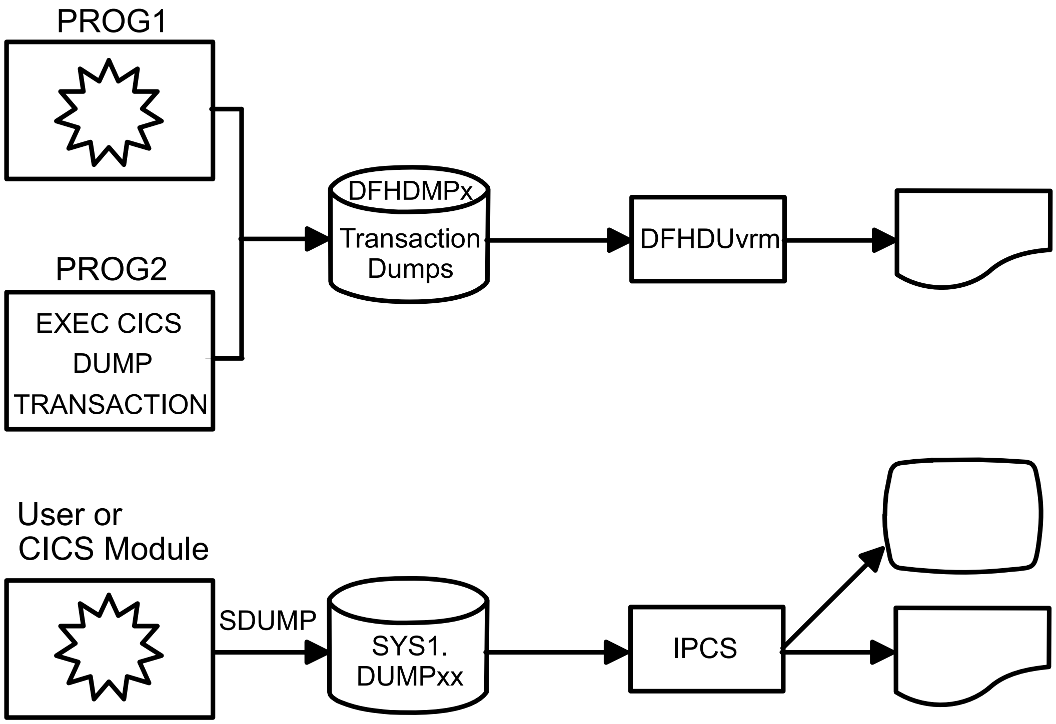


Figure 14-18. Types of Dumps CI207.0

Notes:

- CICS creates a dump when a task abends or when a program requests it.
- Depending on the type of request, one or both kinds of dumps may be created:

Transaction Written to the CICS dump data set and printed with the **Dump Utility Program** `DFHDU640` (CICS TS 3.1) or `DFHDU530` (CICS TS 1.3), respectively.
Usage of the utility is documented in the *CICS Operations and Utilities Guide, SC34-6431*.

System Written to the MVS dump data set and printed with **IPCS**.

The release-specific dump formatting routine `DFHPD640` (for CICS TS 3.1), contained in the SDFHLOAD installation library, must be defined to IPCS by use of MVS PARMLIB members. For details, see the *CICS Transaction Server for z/OS Version 3.1 Installation Guide, GC34-6426*.

Creating Dumps

Cause	Type Dump: Transaction	Type Dump: System
Program check in user task	ASRA	AP0001
Recoverable MVS abend in user task	ASRB	AP0001
Storage protection exception	ASRA	SR0001
Transaction isolation exception	ASRA	SR0001
Subsequently detected storage violation	✓	SM0102
Other task abends	✓	○
EXEC CICS ABEND	✓	○
CEMT PERFORM DUMP	○	MT0001
CEMT PERFORM SHUTDOWN DUMP	○	✓

Figure 14-19. Creating Dumps CI207.0

Notes:

Here are some of the more common causes of dumps.

- A check mark indicates that a dump will normally occur, but the dump code will vary.
- A system dump can be forced or suppressed along with a transaction dump.

Managing the Dump Data Sets

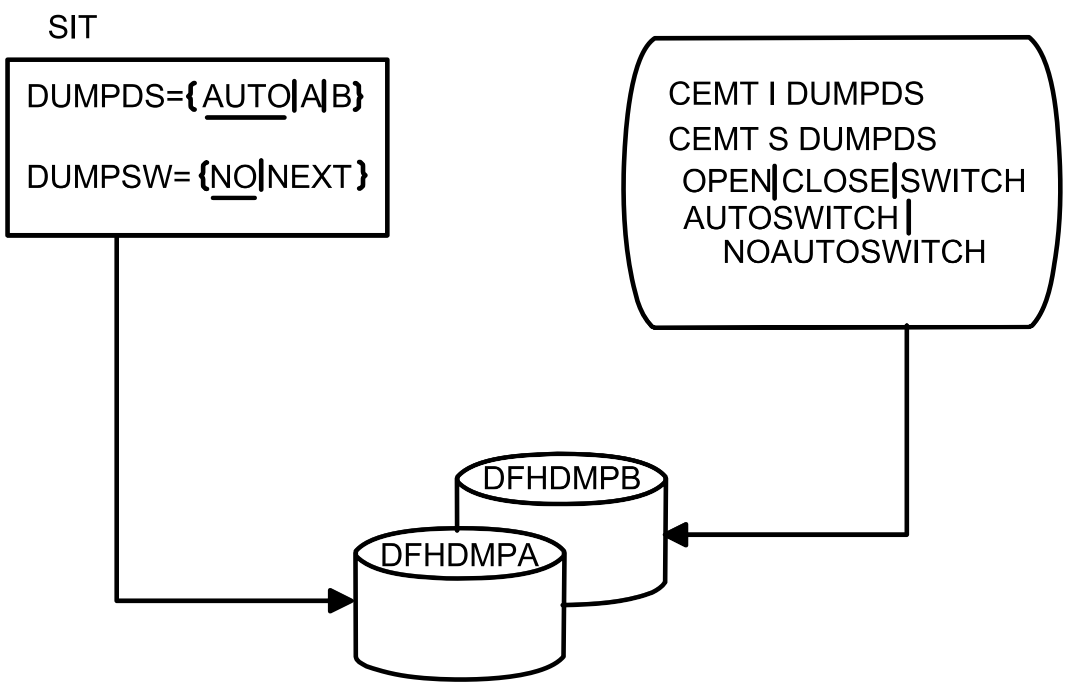


Figure 14-20. Managing the Dump Data Sets CI207.0

Notes:

- Use CEMT I DUMPDS to determine which data set is active. Then overtype to open, close, or set autoswitch for the next time the current data set fills.
- Autoswitch is only good for the next time the data set fills (one time only).
- Enter CEMT S DUMPDS SWITCH to request an immediate switch.

Controlling Dumps

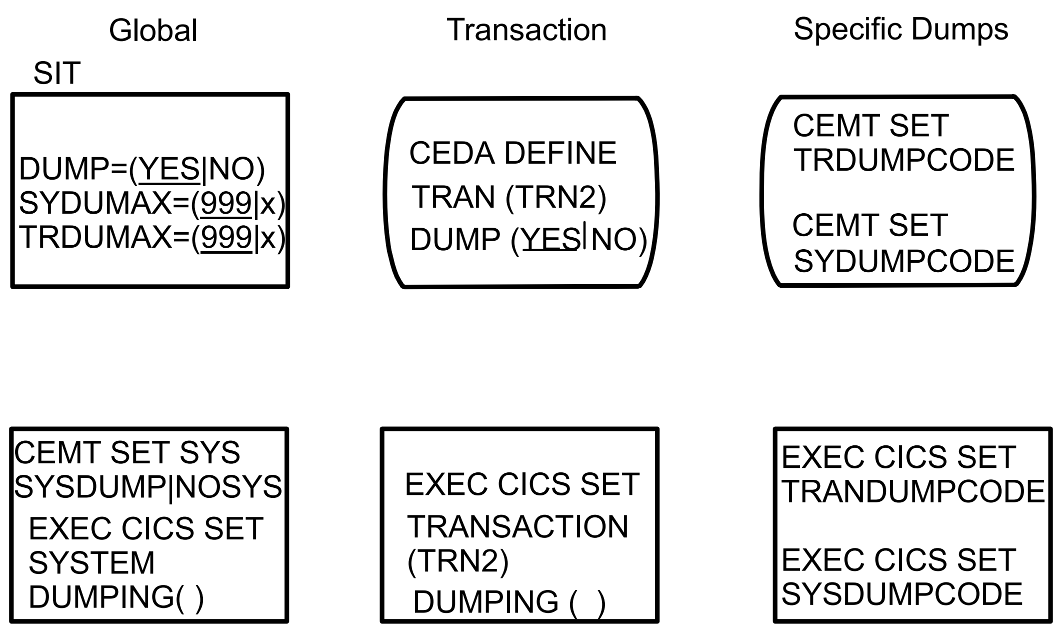


Figure 14-21. Controlling Dumps CI207.0

Notes:

- SDUMPing can be enabled or disabled globally for the CICS region.
- By default, CICS produces a dump for any transaction abend, unless DUMP(NO) is specified in the transaction definition.
- The CEMT transaction allows for building a dump table in order to suppress selected dumps.

Suppressing Dumps

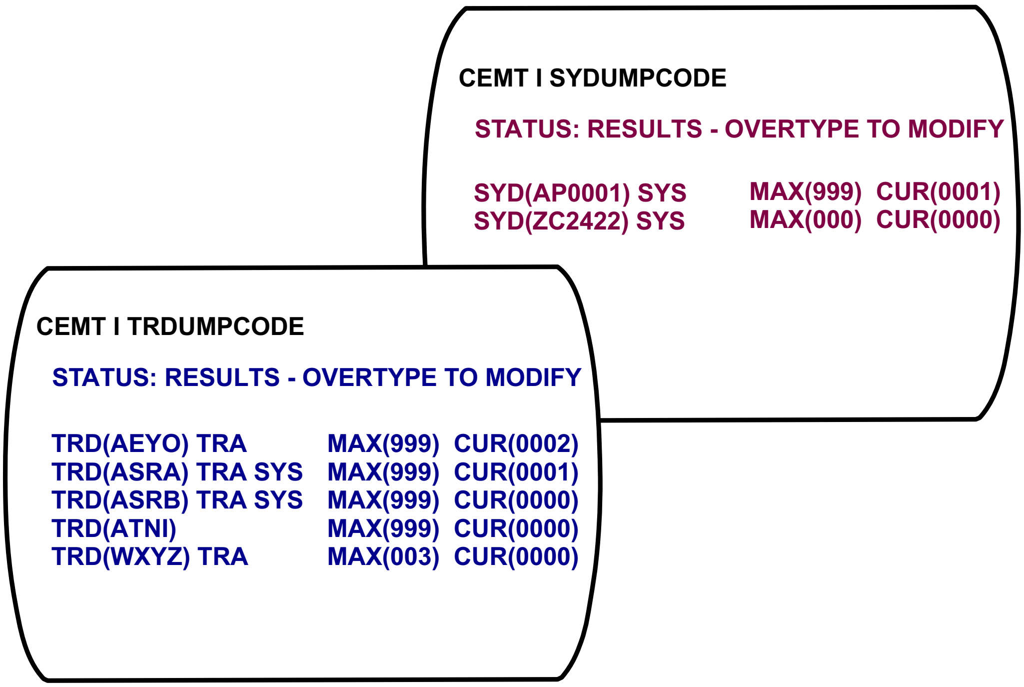


Figure 14-22. Suppressing Dumps CI207.0

Notes:

Suppress or limit dumps with the CEMT command or with EXEC CICS SET TRANDUMPCODE or SYSDUMPCODE. In the example above:

AEY0 Entry generated by CICS; two dumps have occurred.

ASRA Allow both transaction dump and system dump (the default).

ASRB No abend ASRB occurred or CUR value was reset.

ATNI No dumps. ATNI indicates a terminal error, and information in the CSMT log is normally sufficient.

WXYZ Suppress transaction dumps after the first three.

AP0001 Entry generated by CICS; one dump occurred preceding the ASRA.

ZC2422 No system dump taken after error message DFHZC2422E.

Processing Dumps

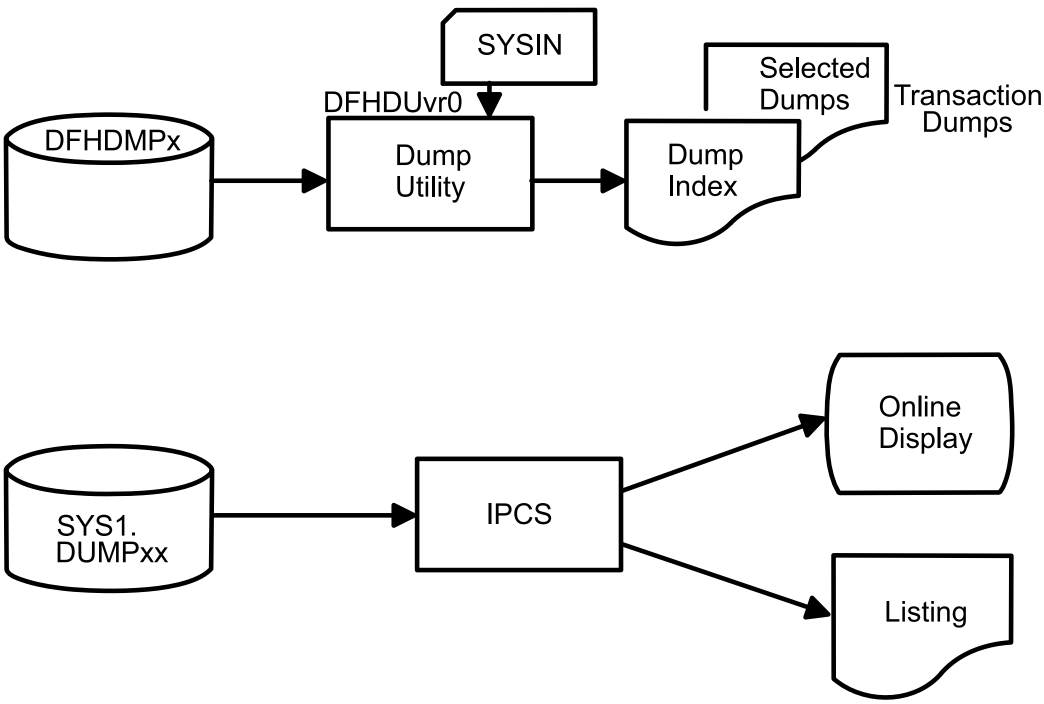


Figure 14-23. Processing Dumps CI207.0

Notes:

- Use the Dump Utility (`DFHDU6x0`) to print the CICS dump data set, selecting which dumps you want based on combinations of transaction code, dump code, and time of day.
- The dump index is a summary of the dumps in the data set. It is used to decide which dumps to print.
- System dumps can be viewed or printed with IPCS, which is invoked through ISPF dialogs.

Basic Problem Determination (Summary)

Figure 14-24. Basic Problem Determination (Summary) CI207.0

Notes:

- The major sources of problem determination assistance provided by CICS are:
  - Messages
  - Transactions
  - Traces
  - Dumps.
- A strategy is needed for capturing and viewing CICS messages directed to transient data.
- CICS can record trace information in an internal table, an auxiliary file, or MVS GTF.
- You can exercise very specific control over what is traced, where it goes, and what transactions may use trace facilities.
- CICS supports two types of dumps: transaction and system.
- Certain dump codes may be suppressed using CEMT.
- System dumps are analyzed by use of IPCS.

Unit Summary

Having completed this unit, you should be able to:

- Outline a simple approach to problem determination
- List the major sources of debugging information
- Select the proper method of handling CICS messages
- List the various types of traces available in CICS
- State CICS specifications for controlling and printing traces
- State CICS specifications for controlling and processing dumps

Figure 14-25. Unit Summary CI207.0

Notes:

Unit 15. Measurement and Evaluation

What This Unit Is About

This unit shows in its first topics the types of information available to help measure and tune CICS TS and how to obtain the data.
The fourth topic briefly introduces the CICS tools that have been introduced in the past years.

What You Should Be Able to Do

After completing this unit, you should be able to:

- Discuss the types of performance data needed to tune CICS TS
- Describe the difference between statistics and monitor information
- Define requirements necessary to collect and print CICS TS performance data

References

CICS System Definition Guide, SC34-6428

CICS Operations and Utilities Guide, SC34-6431

CICS Performance Guide, SC34-6452

Unit Objectives

After completing this unit, you should be able to:

- Discuss the types of performance data needed to tune CICS TS
- Describe the difference between statistics and monitor information
- Define requirements necessary to collect and print CICS TS performance data

Figure 15-1. Unit Objectives CI207.0

Notes:

15.1 Getting Performance Data - Overview

Response Time

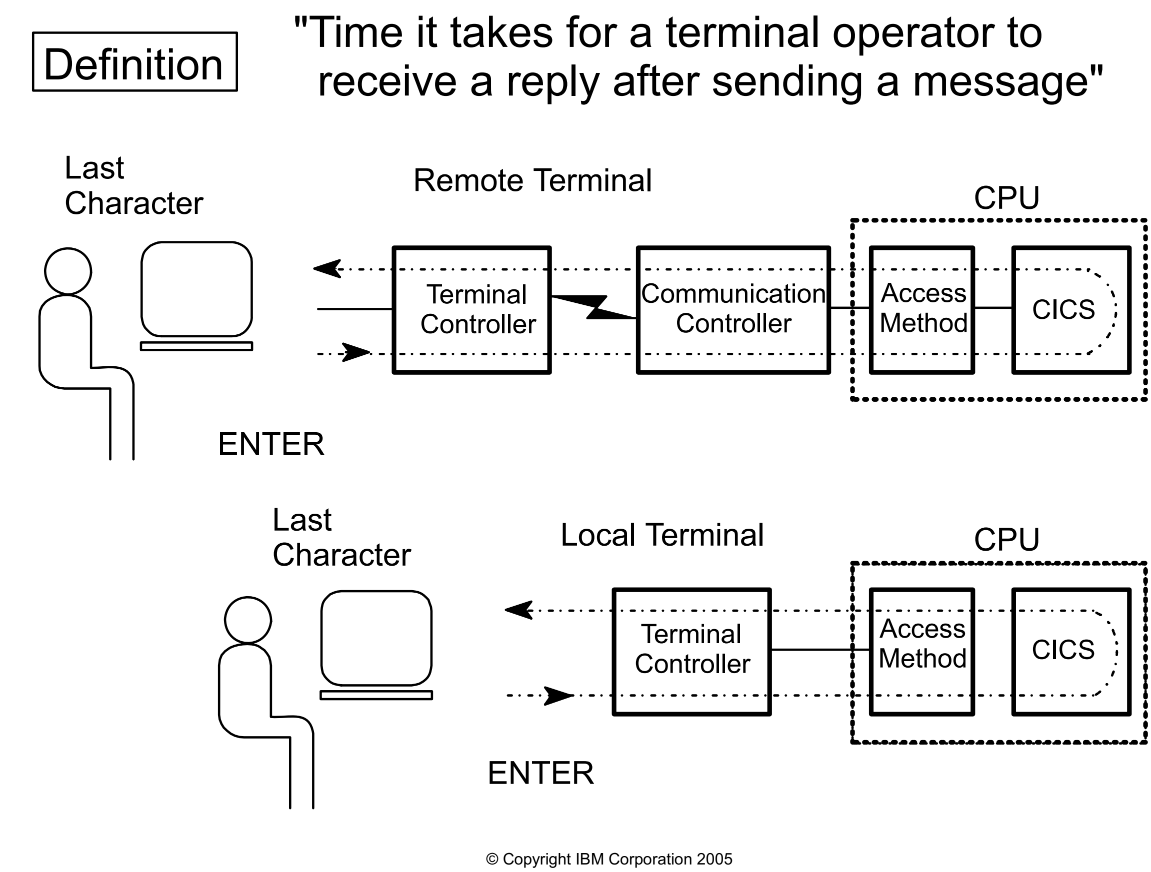


Figure 15-2. Response Time CI207.0

Notes:

- Performance is usually measured by response time.
- You need to understand the network to evaluate response time.
- Response time is not just the time in CICS.

Major Resources Used by CICS

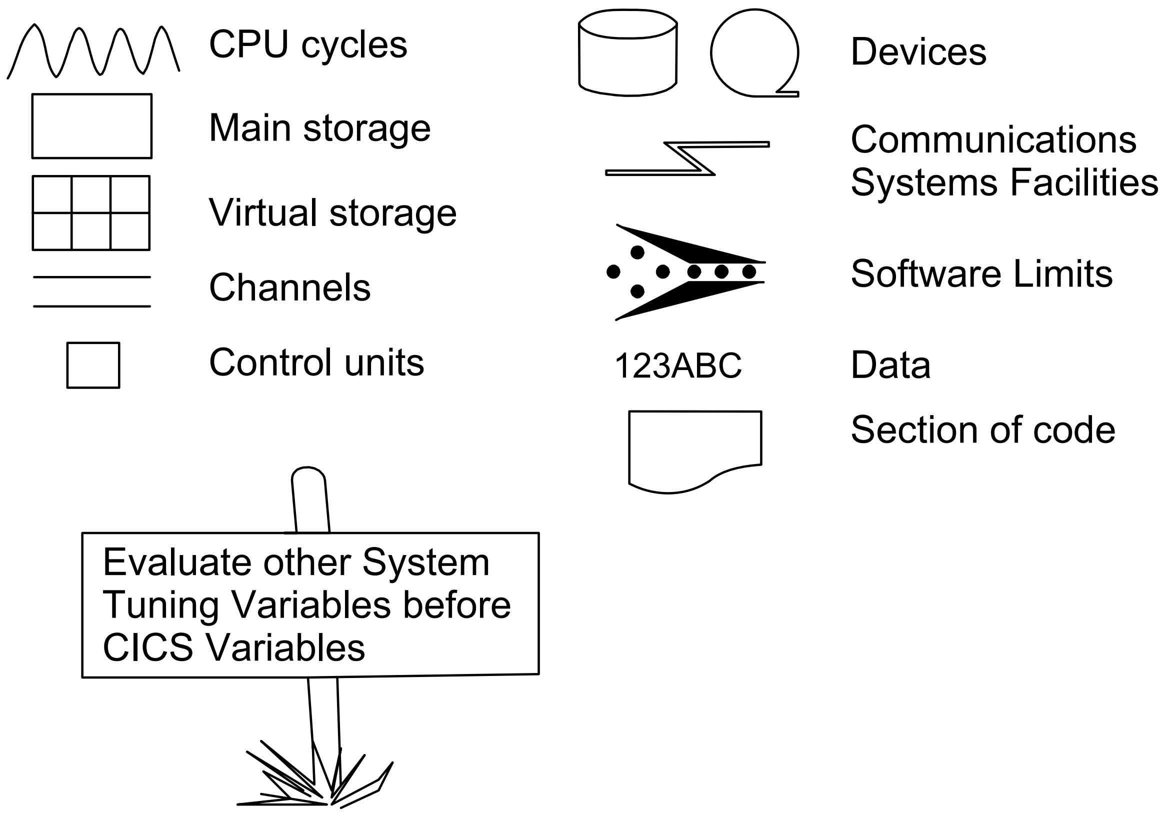


Figure 15-3. Major Resources Used by CICS CI207.0

Notes:

- A resource is a facility used during execution that has a finite capacity.
- When a resource has reached its capacity, CICS response time will be affected.
- Some examples of used resources are:
  - Hardware
  - Software: maxtasks, dsalim, ...
  - Updating of data
  - Single thread code
- Evaluation of all these areas is necessary to meet your response time objectives.

Resource Contention Hierarchy

CPU cycles

↑

Real storage

↑

Virtual storage

↑

Software (specification limits)

↑

Channels

↑

Control units

↑

(Lines)

↑

Devices

Figure 15-4. Resource Contention Hierarchy CI207.0

Notes:

- Contention at the lower levels will prevent full utilization of higher-level resources.
- Contention may be from outside the CICS address space.
- As problems at the lower levels are corrected, higher levels may become a problem.
- Resource contention needs to be balanced for consistent response.

CICS Measurement Tools

Purpose

- Gather data and report on resource availability and use; plus activity for CICS

Examples of Items Reported

- CICS requests
- Response time
- DSA use
- String Use
- Class Maximum Slots
- MXT

CICS Provided Tools

- CICS Statistics
- CICS Monitoring Facility
- CICS provided transactions
- CICS Auxiliary Trace

z/OS Tools

- RMF
- Tivoli

IBM CICS Tools

- CICS PM Performance Monitor
- CICS PA Performance Analyzer
- CICS IA Interdependency Analyzer

Figure 15-5. CICS Measurement Tools CI207.0

Notes:

- These tools, along with others, are described in the *CICS Performance Guide, SC34-6452*.
- Statistics output is also described in the *CICS Performance Guide, SC34-6452*.
- Data produced by the CICS monitor is described in the *CICS Customization Guide, SC34-6429*.
- CICS-supplied transactions allow for controlling the type and frequency of data collection.
- Auxiliary trace data may be used to determine the flow of a task through CICS and the resources the task accesses.
- Recently, a set of new CICS tools has been made available to support CICS customers in the areas of monitoring their CICS systems.

Creating Reports

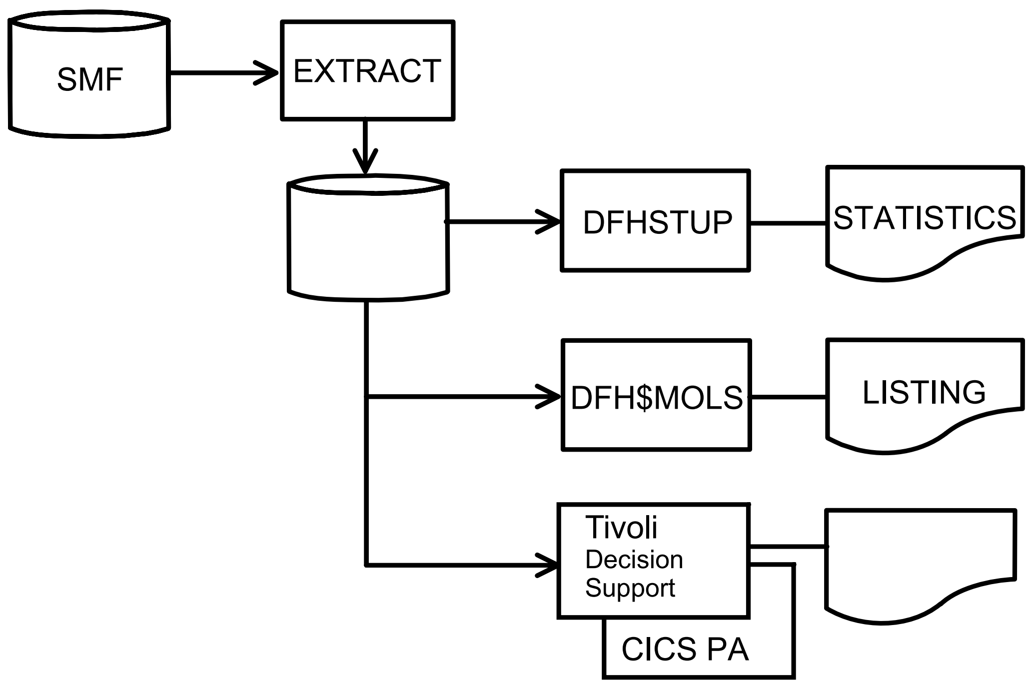


Figure 15-6. Creating Reports CI207.0

Notes:

- Both statistics and monitor data are written to SMF as type 110 records.
- EXTRACT is an SMF utility that lets you select types of SMF records written to a sequential data set.
- After selecting the CICS records, use:

  DFHSTUP CICS-supplied utility that creates a statistics report. Types of reports required may be selected by applid using control cards. The *CICS Operations and Utilities Guide, SC34-6431*, has a sample job stream and control card documentation.

  DFH$MOLS This sample program in SDFHSAMP prints monitor data. Since it only lists data, modifications will be necessary to create specific reports.

- Tivoli Decision Support can process CICS SMF records to produce joint reports with data from other SMF records, for information detailing the relationship between CICS and Tivoli Decision Support.
- CICS Performance Analyzer product is covered by some foils in the next unit.

Statistics versus Monitoring

- Statistics
  - Controlled by resource type
  - Count
  - Time of some requests
- Monitoring
  - Collected by tasks
  - Count
  - Time

Figure 15-7. Statistics versus Monitoring CI207.0

Notes:

- Statistics contain information on the CICS system as a whole, without regard to tasks.
- The CICS monitoring facility collects information about individual tasks.

15.2 CICS TS Statistics

Types of Statistics

Type	When	STATRCD	Reset
Interval	INTERVAL	ON	Yes
End-of-day	ENDOFDAY	——	Yes
Unsolicited	Event-driven	——	No
Requested	Terminal or Program request	——	No
Requested reset		—	Yes

Figure 15-8. Types of Statistics CI207.0

Notes:

- The types of statistics are:

Interval Taken when a specified interval of time has expired. The default is every three hours from midnight.

End-of-day Data collected since the last reset. Information is collected at:

1. End of day - (midnight)
2. Shutdown - (normal or immediate)

Unsolicited Gathered for dynamically allocated and deallocated resources.

Requested User requested by either a CEMT or EXEC CICS.

Requested reset CEMT or EXEC CICS with the RESETNOW option.

- Interval statistics require `STATRCD=ON` in the SIT; the default is OFF.

Programmable Interface

- EXEC CICS PERFORM STATISTICS
- EXEC CICS COLLECT STATISTICS
- EXEC CICS SET STATISTICS

Figure 15-9. Programmable Interface CI207.0

Notes:

- `PERFORM STATISTICS` records statistics for the named resources.
- `EXEC CICS COLLECT` returns either the current statistics for a single resource or global statistics for a resource type to your application program. COPY books are supplied for COBOL, Assembler, and PL/I, defining the format of statistics data.
- `DFH0STAT`, a program in SDFHSAMP, shows how to use `COLLECT STATISTICS`. This program may also be used to analyze storage requirements. The *CICS Operations and Utilities Guide, SC34-6431* describes how the program may be used.
- You may change the `STATRCD`, `INTERVAL`, and `ENDOFDAY` values by the `EXEC CICS SET` command. These may also be changed with CEMT.
- These application commands are documented in the *CICS System Programming Reference, SC34-6435*.
- If command security is implemented, statistics requests are passed to the ESM for authorization.

DFHSTUP Batch Utility Program

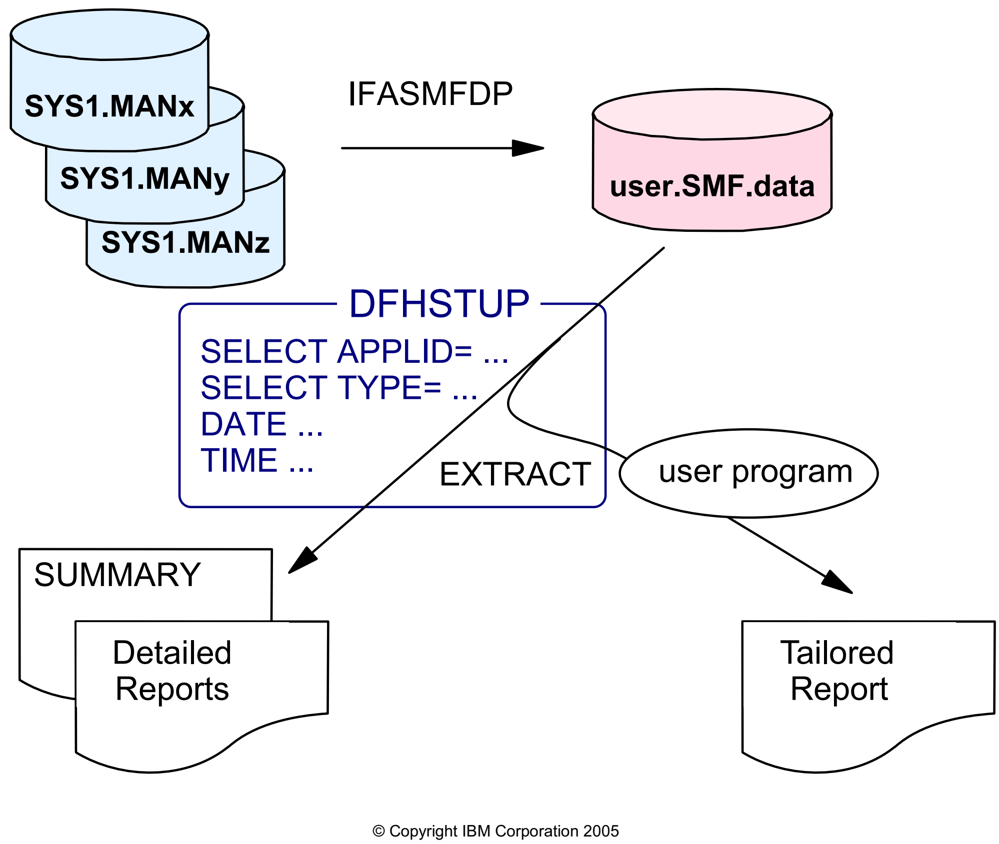


Figure 15-10. DFHSTUP Batch Utility Program CI207.0

Notes:

- The statistics utility program, DFHSTUP, prepares and prints reports offline, using the CICS statistics data recorded on the MVS system management facilities (SMF) SYS1.MANx data sets.
- Control parameters let you specify (among others):
  - The regions (applids) to include or to exclude in your report
  - The resource types to be reported
  - The report interval per date and time
  - Whether or not you like to have a summary report per region
- The EXTRACT function introduced by CICS TS V2.3 allows user-written programs to process statistics records.
  A working sample, DFH0STXR, is provided for event reporting.
- Use the version of the DFHSTUP program from the same release of CICS as the data that it is to process.
- For full syntax and JCL, refer to the *CICS Operations and Utilities Guide, SC34-6431.*

Task Statistic Sample Data

```
Transaction Manager
 Total Accumulated transactions so far. . :            120
 Accumulated transactions (since reset) . :            120     Transaction Rate
 Maximum transactions allowed (MXT) . . . :             20        per second. . :     0.04
 Times at MXT . . . . . . . . . . . . . . :              0
 Current Active User transactions . . . . :              1
 Peak Active User transactions. . . . . . :              3
 Total Active User transactions . . . . . :             82
 Current Running transactions . . . . . . :              1
 Current Dispatchable transactions. . . . :              0
 Current Suspended transactions . . . . . :              0
 Current System transactions. . . . . . . :              0
 Transactions Delayed by MXT. . . . . . . :              0
 Total MXT queuing time . . . . . . . . . :  00:00:00.00000
 Average MXT queuing time . . . . . . . . :  00:00:00.00000
 Current Queued User transactions . . . . :              0
 Peak Queued User transactions. . . . . . :              0
 Total Queuing time for current queued  . :  00:00:00.00000
 Average Queuing time for current queued. :  00:00:00.00000

Dispatcher
 Current ICV time . . . . . . . . . . . . . . :          1,000ms
 Current ICVR time. . . . . . . . . . . . . . :         20,000ms
 Current ICVTSD time. . . . . . . . . . . . . :            100ms
 Current PRTYAGING time . . . . . . . . . . . :            500ms
 MRO (QR) Batching (MROBTCH) value. . . . . . :              1
 Concurrent Subtasking (SUBTSKS) value. . . . :              0
 Current number of CICS Dispatcher tasks. . . :             14
 Peak number of CICS Dispatcher tasks . . . . :             43
 Current number of TCBs attached. . . . . . . :              5
 .....
```


Figure 15-11. Task Statistic Sample Data CI207.0

Notes:

- Transaction Manager statistics
- Transaction Class statistics
- Dispatcher statistics

Storage Management Statistics Sample

```
Storage ABOVE 16MB
 Private Area Region size above 16Mb . . . . . . . . :   1,868,800K
   Max LSQA/SWA storage allocated above 16Mb (SYS) . :      12,804K
   Max User storage allocated above 16Mb (EXT) . . . :     131,936K

 Private Area storage available above 16Mb . . . . . :   1,724,060K
                                              Current EDSA Limit. . . . . . :    122,880K
   CICS Trace table size . . . .       64K    Current Allocation for EDSAs. :     22,528K
   EXT minus Current EDSA Limit.    9,056K    Peak Allocation for EDSAs . . :     22,528K

                                   ECDSA        EUDSA        ESDSA        ERDSA       Totals

  Current DSA Size. . . . . . . :   3,072K      1,024K           0K      18,432K     22,528K
  Current DSA Used. . . . . . . :   2,968K          0K           0K      17,884K     20,852K
  Current DSA Used as % of DSA. :      96%         0%           0%          97%         92% of EDSA Size
* Peak DSA Used . . . . . . . . :   3,000K         64K           0K      17,884K
  Peak DSA Size . . . . . . . . :   3,072K      1,024K           0K      18,432K
  Cushion Size. . . . . . . . . :     128K          0K           0K         256K
  Free Storage (inc. Cushion) . :     104K      1,024K           0K         548K
* Peak Free Storage . . . . . . :   1,052K      1,024K           0K       3,292K
* Lowest Free Storage . . . . . :      72K        960K           0K         548K
  Largest Free Area . . . . . . :      68K      1,024K           0K         516K
  Largest Free Area as % of DSA :       2%        100%           0%           2%
  Largest Free/Free Storage . . :     0.65        1.00         0.00         0.94
  Getmain Requests. . . . . . . :    9,181          14            0          363
  Times no storage returned . . :        0           0            0            0
  Times request suspended . . . :        0           0            0            0
  Times Short-On-Storage. . . . :        0           0            0            0           0
  Storage Violations. . . . . . :        0           0            0            0           0
  Access. . . . . . . . . . . . :     CICS        CICS         CICS     READONLY
0  '*' indicates values reset on last DSA Size change
```


Figure 15-12. Storage Management Statistics Sample CI207.0

Notes:

CICS TS storage manager statistics provide DSA and EDSA summaries, plus detailed information on every DSA storage subpool; the latter is not shown here.

The equivalent report is built for DSAs below 16M.

- CICS **suspends** tasks when a storage request cannot be satisfied. Since this indicates DSA overutilization, significant non-zero values here should be a cause for concern.
- **Task subpool** statistics show total DSA currently allocated to task-related storage.

File Statistics Sample

Files

Filename	Access Method	File Type	Remote Filename	Remote System	LSR Pool	RLS	Recovery Status	Str-ings	<- Buffers-> Index	Data
DFHCMACD	VSAM	KSDS			1	No	NotRecoverable	1	1	2
DFHCSD	VSAM	KSDS			1	No	Recoverable	6	0	0
DFHDBFK	VSAM	KSDS			No	No	NotRecoverable	1	1	2
DFHLRQ	VSAM	KSDS			1	No	Recoverable	10	10	11
FILEA	VSAM	KSDS			1	No	Recoverable	3	3	4

Files - Requests

Filename	Read Get Requests	Update Requests	Browse Requests	Browse Updates	Add Requests	Update Requests	Delete Requests	<--- String Waits ---> Total	HWM
DFHCMACD	3	0	0	0	0	0	0	0	
DFHCSD	0	0	0	0	0	0	0	0	0
DFHDBFK	0	0	0	0	0	0	0	0	0
DFHLRQ	0	0	0	0	0	0	0	0	0
FILEA	0	0	159	0	0	0	0	0	0

Data Set Names

Data Set Name	Access Method	Dsname Object	Dsname Validity	Dsname Availability	File Count	Recovery Status
EDUCI.CTS220.CICSIVP.DFHLRQ		Base	Valid	Available	1	Recoverable
EDUCI.CTS220.CICSIVP.FILEA		Base	Valid	Available	1	Recoverable
EDUCI.CTS220.DFHCMACD		Base	Valid	Available	1	Undetermined
EDUCI.CTS220.DFHCSD		Base	Valid	Available	1	FwdRecoverable

Figure 15-13. File Statistics Sample CI207.0

Notes:

File statistics provide the following information:

- Which files or data sets have been opened and closed at what time during CICS execution (in other words, during the latest statistic interval).
- Which and how many requests have been made against which file.

DFH0STXR Sample Output

```
EXCEPTION
WARN
INFO
CICS 6.3.0 Statistics Exception Event Reporter
Report Date: 08/30/2002
Report Time: 15:51:18      Page      2
________________________________________________________________________________
        Record      Record   Stats          Event
Type     Date        Time    Type  Applid    id    Event Description
________________________________________________________________________________
  I   08/30/2002  15:42:43  REQ  IYK2Z2G1  XM001  MXT Limit =  30.
                                 Current Tasks =   1. Peak Tasks =   6.

  I   08/30/2002  15:42:43  REQ  IYK2Z2G1  SM002  EDSA Limit =    204,800K.
                                 Allocated =     22,528K. Peak =     22,528K.

  I   08/30/2002  15:42:43  REQ  IYK2Z2G1  TS001  DFHTEMP CIs =       507.
                                 Peak Used =       2. Peak TSMain = 0K.

  W   08/30/2002  15:42:43  REQ  IYK2Z2G1  LD001  Program Load to Use Ratio
                                 for Program 'IBMRLIB1' = 100%.
                                 Location = 'RDSA    '.

  I   08/30/2002  15:42:43  REQ  IYK2Z2G1  D2001  DB2 Conn 'RCTJT   '.
                                 TCB Limit =    12. Peak TCBs =     1.
                                 TCB Readyq HWM =   0.
```


Figure 15-14. DFH0STXR Sample Output CI207.0

Notes:

- The EXTRACT statistics reporting function introduced with CICS TS V2R3 is exploited by providing a sample extract exit program, **DFH0STXR**, which produces an event type of statistics report that highlights each event in a single print line. This provides a much easier method of analyzing CICS statistics records than has been available in previous releases.
- There are three types of event:
  - Exception -- a CICS limit condition has been exceeded (such as MXT, or file strings).
  - Warning -- a peak condition has met or exceeded a user-defined threshold percentage.
  - Information -- a peak value has been reached for a key CICS system parameter.

  These events are predefined within the DFH0STXR sample program, but can be customized.
- DFH0STXR is supplied both as COBOL source program (in SDFHSAMP) and as an executable load module (in SDFHLOAD). The visual shows a typical output produced by DFH0STXR (slightly edited to fit on the page).

15.3 Monitoring

Classes of Monitor Data

- Performance
  - Detailed transaction information
  - Elapsed Time
  - Waiting for I/O
  - ...
- Exception
  - Exceptional Conditions
  - Queuing for VSAM string/buffer
  - Waiting on storage
  - Temporary storage exhausted

Figure 15-15. Classes of Monitor Data CI207.0

Notes:

- Monitoring is the CICS facility that collects data for a task.
- The *CICS Performance Guide, SC34-6452*, contains details of the performance and exception data captured.
- Exception data can be used to help identify performance constraints.
- Until CICS/ESA V4, there was another option, SYSEVENT, for measuring internal response times. This has been replaced by support for MVS Workload Manager, as documented in the *CICS Performance Guide, SC34-6452*.

Task Data

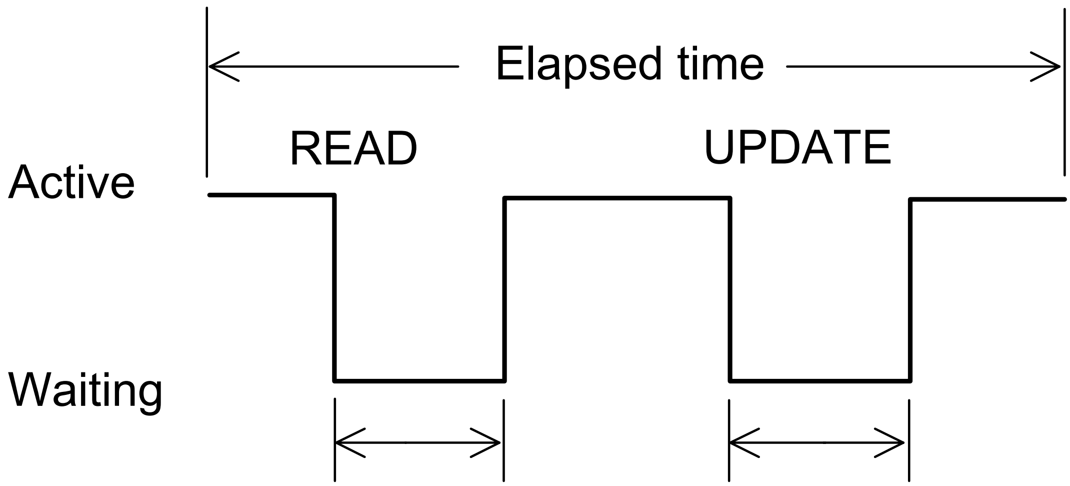


Figure 15-16. Task Data CI207.0

Notes:

- The CICS monitor facility records total task time.
- Elapsed time represents the time of a unit of work in CICS (internal response time).
- The monitor further divides total time into:
  - Dispatch time
  - CPU time, if collecting CPU usage
  - I/O wait time
  - Suspend time

Controlling Data Collection

- SIT
  - MCT=NO|suffix|YES
  - MN=OFF|ON
    MNEXC=OFF|ON
    MNPER=OFF|ON
  - MNCONV=NO|YES
  - MNFREQ=0|hhmmss
  - MNSYNC=NO|YES
- CEMT INQ | SET MONITOR
- EXEC CICS INQ | SET MONITOR

Figure 15-17. Controlling Data Collection CI207.0

Notes:

- Use SIT parameters to initialize CICS with monitor either OFF or ON.
- Specify the type of data to collect.
- With `MCT=NO`, CICS dynamically builds a default Monitor Control Table (MCT).
- Startup options may be changed by either an application program or an operator. If command security is implemented, monitor requests are passed to the ESM for authorization.
- Check the *CICS System Definition Guide, SC34-6428*, for CICS defaults.
- For performance monitoring, you may request:
  - One record per dialog step of a conversational task
  - Records to be written for long-running transactions before task termination periodically after a certain time
  - Records to be written when a syncpoint is issued during task execution

User Monitoring

- Monitor Control Table MCT
  - define event monitoring points (EMP)
    - counters
    - clocks
    - character strings
- Refer to EMP in user programs
  - EXEC CICS MONITOR commands

Figure 15-18. User Monitoring CI207.0

Notes:

- An MCT has to be provided only for user monitoring.
- Documentation for the MCT is in the *CICS Resource Definition Guide, SC34-6430*.
- User EMPs are discussed in the *CICS Application Programming Guide, SC34-6433*.
- EMPs are necessary for DL/I performance data.

  Use SDFHSAMP-provided sample MCT entries DFH$MCTD (for DBCTL).

Measurement and Evaluation (Summary)

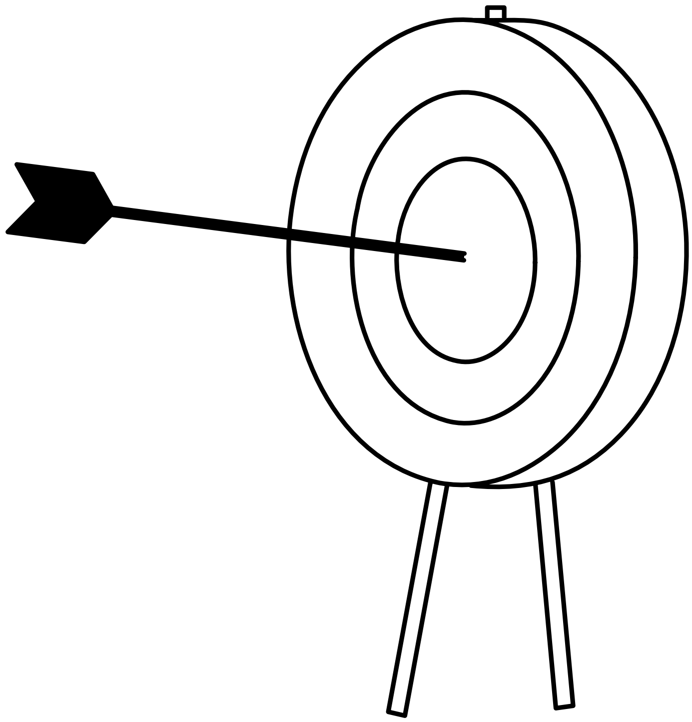


Figure 15-19. Measurement and Evaluation (Summary) CI207.0

Notes:

- Measure performance on a regular interval to assist in predicting when performance problems will occur.
- Statistics are collected by resource type.
- Monitor data is recorded for each task.
- Statistics and monitor collection are controlled with SIT parameters or by CEMT and EXEC CICS commands.
- A MCT is required for cycle information, more meaningful conversational task data, and user EMPs.
- Both statistics and monitoring data go to SMF.
- Utilities or sample programs may be used to print the data.

Unit Summary

Having completed this unit, you should be able to:

- Discuss the types of performance data needed to tune CICS TS
- Describe the difference between statistics and monitor information
- Define requirements necessary to collect and print CICS TS performance data

Figure 15-20. Unit Summary CI207.0

Notes:

Unit 16. Related CICS Program Products (Optional Unit)

What This Unit Is About

IBM provides a range of CICS tools, and other products which support the use of the CICS Transaction Server for z/OS. This unit will provide a brief overview of these products.

What You Should Be Able to Do

After completing this unit, you should be able to:

- **Describe the following CICS tools:**
  - IBM Performance Analyzer for z/OS
  - IBM CICS Online Transmission Time Optimizer
  - IBM CICS Interdependency Analyzer
  - CICS VSAM Recovery
  - IBM Session Manager
  - CICS Business Event Publisher for MQSeries

How You Will Check Your Progress

Accountability:

- Checkpoint questions
- Classroom discussion

References

GC34-6421 *CICS Release Guide*

Information Center

Unit Objectives

After completing this unit, you should be able to describe the CICS tools that help you to manage and control your system:

- IBM Performance Analyzer for z/OS
- IBM CICS Online Transmission Time Optimizer
- IBM CICS Interdependency Analyzer
- CICS VSAM Recovery
- IBM Session Manager
- CICS Business Event Publisher for MQSeries

Figure 16-1. Unit Objectives CI207.0

Notes:

Unit Overview

- IBM Performance Analyzer for z/OS
- IBM CICS Online Transmission Time Optimizer
- IBM CICS Interdependency Analyzer
- CICS VSAM Recovery
- IBM Session Manager
- CICS Business Event Publisher for MQSeries

Figure 16-2. Unit Overview CI207.0

Notes:

16.1 CICS Tools Overview

IBM CICS-Related Program Products (1 of 2)

- IBM performance analyzer for z/OS
  - CICS PA
  - Offline performance reporting tool
  - Analyzes CICS, DB2, MVS System Logger and other subsystems' SMF records
- IBM CICS interdependency analyzer
  - CICS IA
  - Reports CICS resource interdependencies
  - Provides online report queries

Figure 16-3. IBM CICS-Related Program Products (1 of 2) CI207.0

Notes:

These products are priced as one-time charge (OTC).

IBM CICS Performance Analyzer (CICS PA) for z/OS, a powerful offline tool, produces a wide range of reports and extracts that will help customers tune and manage their CICS systems.

IBM CICS Online Transmission Timer Optimizer (CICS OTTO) for z/OS compresses data streams for 3270, SCS printers, 3600 and 4700 systems.

IBM CICS Interdependency Analyzer (CICS IA) for z/OS helps to identify resource interdependencies within CICS systems, and provides information which helps to take advantage of the workload balancing capabilities of CICS TS and to build more flexible CICS systems to increase productivity.

IBM CICS-Related Program Products (2 of 2)

- CICS VSAM Recovery
  - CICSVR
  - Recover lost or damaged VSAM data sets
- IBM Session Manager
  - Access multiple VTAM and TCP/IP terminal sessions
- CICS Business Event Publisher for MQSeries
  - CICS BEP
  - Creates a MQ message from a CICS event

Figure 16-4. IBM CICS-Related Program Products (2 of 2) CI207.0

Notes:

CICS VSAM Recovery (CICSVR) recovers lost or damaged VSAM data sets for OS/390 and z/OS customers. CICSVR can help users maintain 24-hour availability of their data and ensure its integrity.

IBM Session Manager for z/OS is a session manager for VTAM and TCP/IP that provides a secure and user-friendly method of accessing multiple OS/390 or z/OS systems from a single 3270 terminal.

CICS Business Event Publisher for MQSeries (CBEP) creates MQSeries messages based on user-selected events within CICS applications.

16.2 CICS PA

CICS PA

- **IBM CICS Performance Analyzer for z/OS**
- **Comprehensive performance reporting for CICS**
- **Allows user field selection**
- **Not an online monitor - batch reports only**
- **Uses the following input data:**
- **CICS Monitoring Facility - (SMF 110)**
- **DB2 Accounting data - (SMF 101)**
- **MVS System Logger - (SMF 88)**
- **MQSeries Accounting data - (SMF 116)**
- **Complements existing CICS utilities**
- **DFH$MOLS, DFHSTUP and DFH0STAT**
- **Program product: 5655-F38**

Figure 16-5. CICS PA CI207.0

Notes:

CICS PA is a powerful offline reporting tool, and produces a wide range of reports and extracts that will help you tune and manage your CICS systems.

Some of the reports are:

- Transaction response time
- CICS system resource usage
- Cross system performance

External subsystems include DB2, IMS, and MVS System Logger.

CICS PA Reports and Extracts

CICS PA provides an ISPF menu-driven dialog to help you request and submit your reports and extracts.

Performance Reports:	**Exception Reports:**
List	**List**
Summary	
Performance Graph Reports:	**Performance Extracts:**
Transaction Rate	**Cross-system Work**
Transaction Response Time	**Export**
	Record Selection
List Extended	
Summary	
Totals	
Cross-System Work	
Transaction Group	
BTS	
Workload Activity	
DB2	
System Logger	
Transaction Resource Usage	

Figure 16-6. CICS PA Reports and Extracts CI207.0

Notes:

The CICS PA dialog automatically generates the commands and JCL for batch report processing.

CICS Performance Analyzer PA

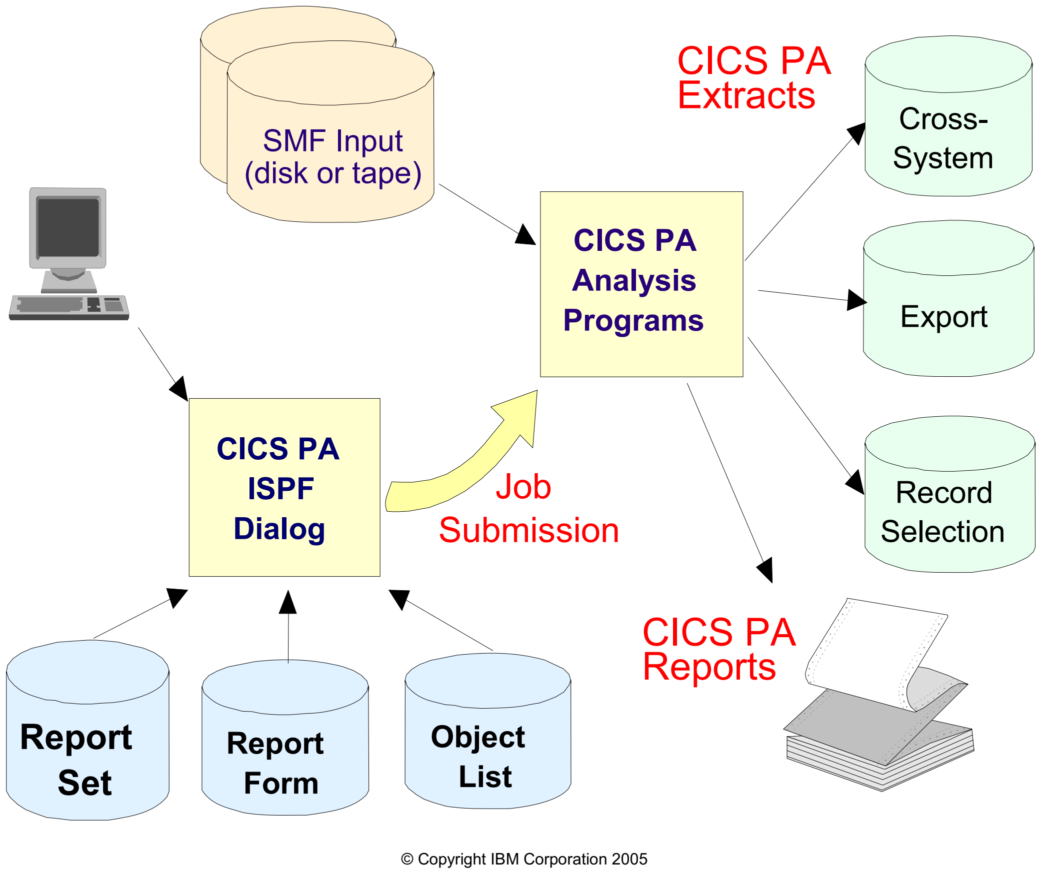


Figure 16-7. CICS Performance Analyzer PA CI207.0

Notes:

This visual shows the main components of CICS PA, including the TSO ISPF dialog, its related data sets, and the CICS PA batch analysis and reporting programs.

The CICS PA analysis programs use the performance and accounting data written to MVS System Management Facility (SMF) data sets. This includes the data collected by the CICS Monitoring Facility (CMF) and written as SMF type 110 records, DB2 Accounting data written as SMF type 101 records, by the MVS System Logger written as SMF type 88 records, and MQSeries type 116 records.

You can produce all the CICS PA reports and extracts by simply defining your CICS Systems (APPLIDs), MVS images, DB2 subsystems, MVS System Logger, and their associated unloaded SMF data sets.

CICS PA data sets include:

- Report sets, to define your report and data extract requests.
- Report forms, to enable you to tailor your reports and extracts to include the information that you want to see.

- Object lists, to enable you to group objects for reporting purposes, for example, to analyze the resource usage of a particular group of transactions or users.

Comprehensive online help is available for every CMF field, so that you never need to reference a manual.

Export is the way to output record separated data in a customized format to your PC for input to other tools, for example Lotus 1-2-3, Lotus Approach, or Microsoft Excel.

So CICS PA can help to:

- Analyze CICS application performance
- Improve CICS resource usage
- Evaluate the effects of CICS system tuning efforts
- Improve transaction response time
- Provide ongoing system management and measurement reports
- Increase availability of resources
- Increase the productivity of system and application programmers
- Provide awareness of usage trends
- Assisting future growth estimates

CICS PA also provides:

- Full support for the latest CICS TS V2 enhancements (including EJB, ECI over TCP/IP, and so forth)
- Transaction performance across IMS DL/I, DB2, and MVS System Logger in a single report
- Detailed MVS System Logger performance reports

16.3 CICS OTTO

CICS OTTO

- **CICS Online Transmission Time Optimizer**
  - **CICS specific solution (not VTAM)**
  - **CICS OTTO optimizes:**
    - **Data streams directed to 3270-type display stations and/or printers**
    - **Data streams directed to SCS-type printers**
    - **Data streams directed to banking terminals 3600/4700**
  - **Runtime tool**
  - **ISPF user interface**
  - **Easy to install, customize, and use**
  - **Program product: 5655-I05**

Figure 16-8. CICS OTTO CI207.0

Notes:

IBM CICS Online Transmission Time Optimizer (CICS OTTO) improves 3270 network resource utilization and response time, and increases user productivity by performing the following basic functions, all of which are transparent to users and applications:

- Repetitive characters — typically as much as 25 percent of all characters sent to terminals and other 3270 network devices — are reduced to only four bytes.
- Outbound data transmission to the terminals is minimized by keeping screen layout in memory and removing data fields already present on the screen.
- Blank spaces are eliminated to improve print speed.

CICS OTTO is a CICS-specific tool, that is, it is not a VTAM solution.

CICS OTTO Optimization Techniques

- Exit to DFHZCP
- Repetitive character elimination for 3270-type terminals and printers
- Transmission of only changed data for 3270-type terminals
- Keeps an image of the actual screen layout in virtual storage
- Blank elimination for 3270 SNA Character Set (SCS) printers
- String Control Byte (SCB) compression for 3600/4700-type terminals
- Outbound and inbound

Figure 16-9. CICS OTTO Optimization Techniques CI207.0

Notes:

CICS Online Transmission Time Optimizer for z/OS helps identify and remove repetitive data by examining and dynamically compressing outgoing data streams.

Typically, as much as 25 percent of all characters sent to terminals and other 3270 network devices are reduced to only four bytes, reducing transmitted message size considerably.

CICS OTTO also minimizes outbound data transmission to the terminals by keeping screen layout in memory and removing data fields already present on the screen.

Blank spaces are eliminated to improve printing speed.

CICS OTTO ISPF Interface

- **Control all optimization features, run traces, and manage system statistics**
  - **Provided sub-functions**
  - **Start or stop component type**
  - **Display and control the image pool size**
  - **Select or exclude specific terminals or modules to optimize**
  - **Add or move terminals**
  - **Start and stop trace**
  - **Display statistics**
  - **Exclude terminals from optimization**

Figure 16-10. CICS OTTO ISPF Interface CI207.0

Notes:

The ISPF user interface allows you to control all optimization features, run traces, and manage system statistics. Some of its subfunctions are:

- Start or stop CICS OTTO for each component type
- Display and control the image pool size
- Select or exclude specific terminals or modules to optimize
- Dynamically add or remove terminals or modules from optimization in runtime
- Start and stop trace
- Display statistics
- Create an exclude list to limit optimization on certain logical units because the session is used to transfer a file

16.4 CICS IA

CICS IA

- IBM CICS Interdependency Analyzer for z/OS
  - Resource interdependencies
  - What CICS region has it?
  - What resource does a transaction need to run?
  - Which program uses which resources?
  - Online and batch reports
  - Reports data stored in a DB2 database
  - Output can be used by CPSM's workload manager
  - Program Product: 5655-G76

Figure 16-11. CICS IA CI207.0

Notes:

CICS IA is a runtime tool that helps you identify resource interdependencies within a CICS system using report data stored in a DB2 database. You can query the reports data online, which will help you to improve your ability to maintain, enhance, and migrate your business applications.

This tool gives the system programmer and managers necessary knowledge to:

- Split workload and move applications to more CICS regions
- Exploit the benefits of workload balancing across the CICSPlex

CICS IA - Overview

- Collector
  - Gathers information on resource interdependencies
  - Staged to VSAM file through data spaces for reduced overhead
- Dependency Database
  - Contains accumulated data about all your applications and the resources that they use
  - Contains user-defined applications using SQL statements
- The Query Interface
  - CICS COBOL BMS programs that dynamically interrogate the Dependency Database
- Scanner
  - Analyzes the members in the Application LoadLib to produce
    - Summary Reports
    - Detailed Reports
- Reporter
  - Analyzes the interdependency data stored on VSAM file
  - Selectable by Resource type

Figure 16-12. CICS IA - Overview CI207.0

Notes:

- **Collector** - Gathers information on resource interdependencies, and staged to VSAM file through data spaces for reduced overhead.
- **Dependency Database** - Contains accumulated data about all your applications and the resources that they use. The database can be updated from VSAM files under operator control.
- **The Query Interface** - Is the suite of CICS COBOL BMS programs that dynamically interrogate the Dependency Database by resource type; for example, list all the files used by application.
- **The Scanner** - Analyzes the members in the Application LoadLib to produce: summary reports which show module name, length, language, number of dependency commands; and detailed reports which shows details including offset, storage content, possible command and dependency type.
- **Reporter** - Analyzes the interdependency data stored on VSAM. The data is selectable by resource type and shows transaction, program, program offset, command, resource name, SYSID and usage.

CICS Interdependency Analyzer IA

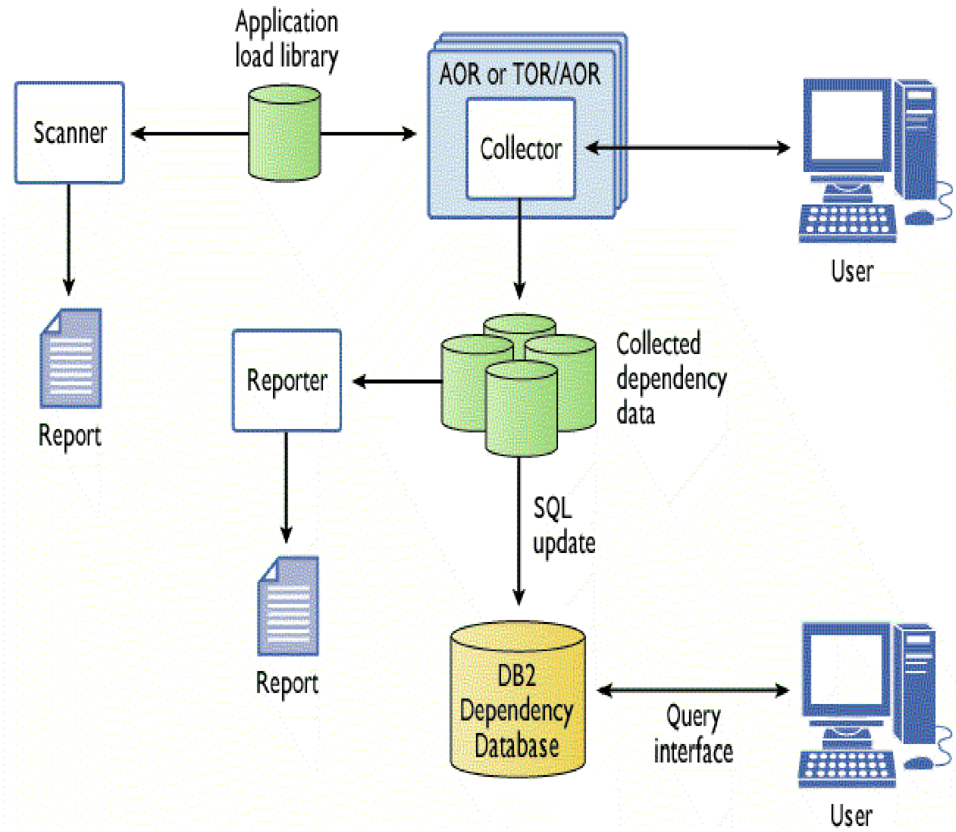


Figure 16-13. CICS Interdependency Analyzer IA CI207.0

Notes:

CICS IA automates the process of collecting the data on the interdependencies between CICS resources. CICS IA works in two ways: offline and online.

- Online components:

  The **Collector** intercepts CICS commands which may be involved in an interdependency and records the details of the resources used. For efficiency's sake, the data is stored in a data space, which, under user control, may be offloaded to the Collected dependency data component. These data sets may be aggregated together and stored in a DB2 database.

- Offline components:

  The **Reporter** presents the dependency information collected by the Collector for a selected CICS region, in a structured format. The scanner provides the additional capability to scan the load module data sets detecting EXEC CICS commands that may cause transaction resource dependency, and to produce a printed report.

So, the overall benefits of CICS IA are:

- It helps understand resource usage and relationships in the CICS applications.
- It assists workload management.
- It Improves your ability to maintain and enhance existing applications.
- It helps to plan reuse of existing applications as e-business applications.

CICS IA - Which Resources Are in an Application?

```
CICS IA V1.1  CICS Interdependency Analyzer for z/OS and OS/390              \

In      Tran  Program       Links/XCTL             Strts    File     Map/Set * * * *
Regn                        Loads         Tran
CORD    VA10  CAMA100C      CAMA120C
        VA10  CAMA100C      CAMA800C
        VA10  CAMA100C                                      CAM1001
        VA10  CAMA120C                             EZPS21


PF3=End   PF4=Exit  PF7=Up    PF8=Down
```


Figure 16-14. CICS IA - Which Resources Are in an Application? CI207.0

Notes:

The above display shows the results of a query: What programs, map set, and file does transaction VA10 access?

The report has been modified to fit on the page.

16.5 CICSVR

CICSVR

- CICS VSAM Recovery provides automated recovery of damaged data sets
- Functions provided:
  - VSAM Forward Recovery
  - Backup-while-open
  - Logstream copy utility
  - CICSVR Server address space
  - Batch logging
  - Log stream change accumulation processing
  - VSAM Record Level support
  - CICSVR ISPF dialogs
  - Management of VSAM Spheres
  - Generate recovery JCL
  - Remote Recovery site support
  - Online Help
  - Recovery Reports
- Program product: 5655-H93

Figure 16-15. CICSVR CI207.0

Notes:

IBM CICS VSAM Recovery, Version 3 Release 2 recovers your lost or damaged CICS and batch VSAM data. CICSVR is for those organizations for whom availability and integrity of CICS VSAM data are vital.

The CICSVR forward recovery function helps you recreate a VSAM sphere from a backup copy of the sphere. Use the panel interface to restore a DFSMShsm or DFSMSdss backup, reapply all changes made to the VSAM sphere since the last backup, and return the VSAM sphere back to the exact state before the data was lost.

Log stream copy utility - You can use the log stream copy utility to copy an MVS log stream to up to nine sequential access method (SAM) data sets. You cannot use IDCAMS or CICSVR archive to copy an MVS log stream; you must use the CICSVR log stream copy utility to copy a log stream.

In order to perform batch logging or change accumulation, the CICSVR server must be active.

CICSVR ISPF panels allow you to select which VSAM data sets you wish to recover and which backup data sets to use. New CICSVR commands will assist you in a remote disaster recovery site by allowing control data sets to be exported and imported.

CICSVR Overview

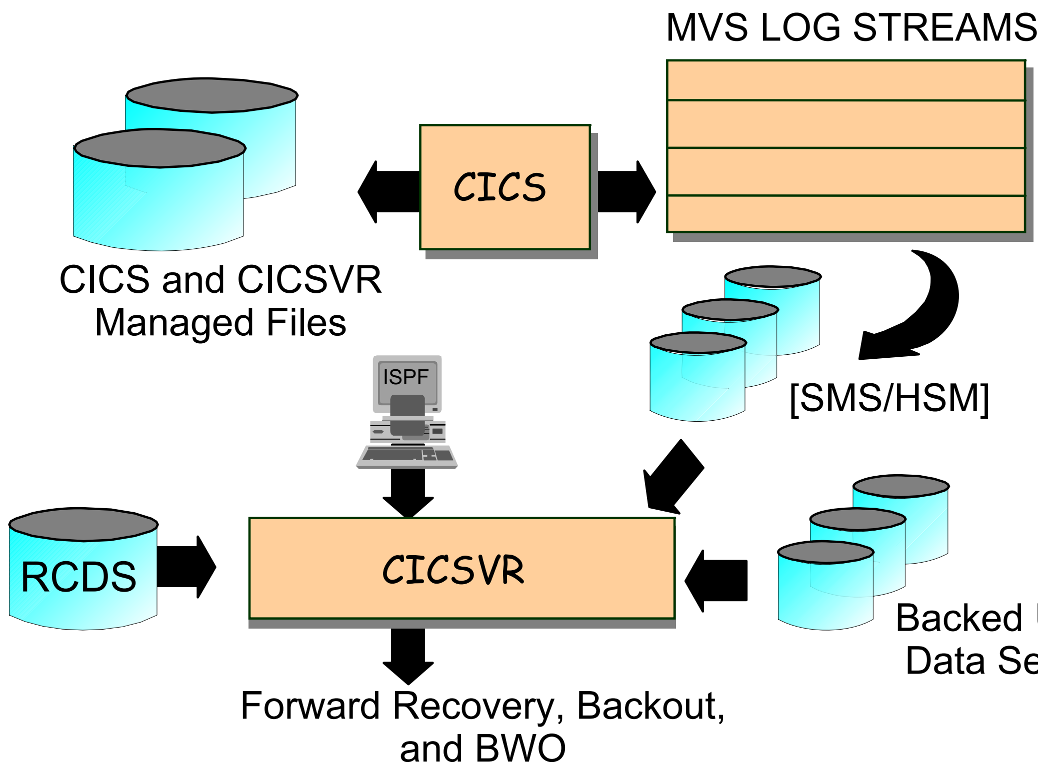


Figure 16-16. CICSVR Overview CI207.0

Notes:

RCDS stands for Recovery Control Data Sets: one of three identical linear VSAM data sets that contain information about the contents of archived logs and the ISPF dialog interface default values. CICSVR uses this stored information to construct recovery jobs.

VSAM Sphere

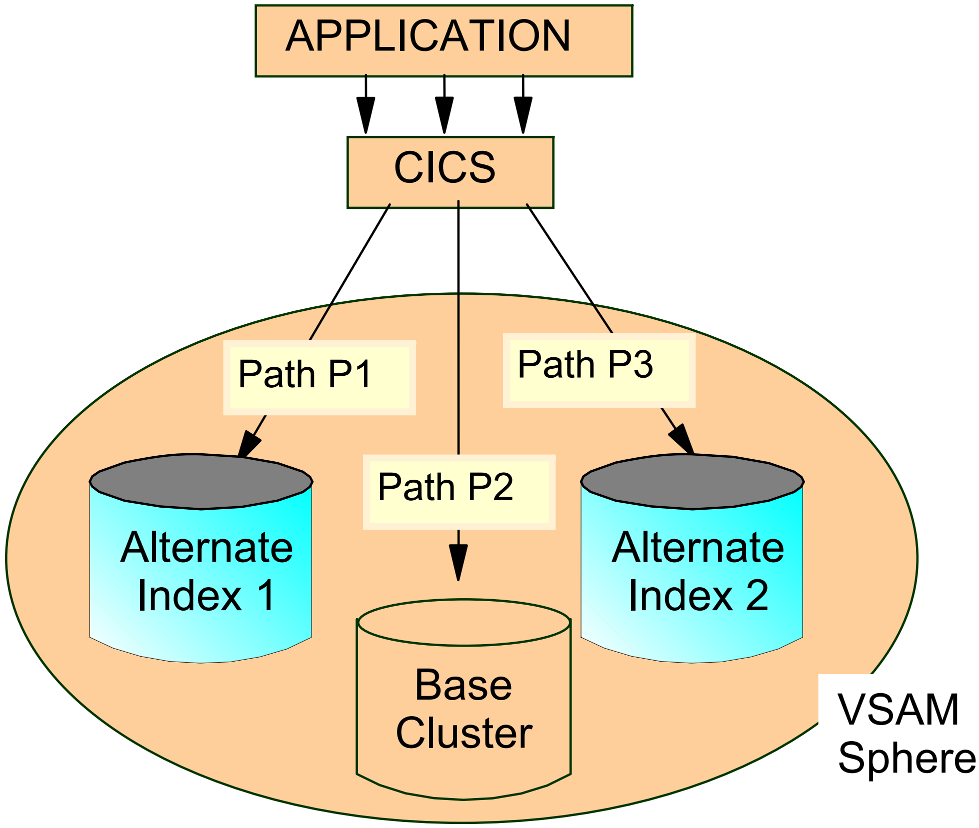


Figure 16-17. VSAM Sphere CI207.0

Notes:

VSAM Sphere is:

- A CICSVR basic unit of recovery
- A VSAM base cluster, containing data
- Optionally, one or more alternate index data sets that contain record keys of data records in the base cluster

In CICSVR, several logically-related VSAM spheres can be grouped into a VSAM sphere group. For example, a payroll application consists of many files, including personnel, expense, and salary files. To maintain logical integrity, backup and recovery operations should be performed simultaneously on all of them.

16.6 IBM Session Manager

IBM Session Manager

- Provides secure access to multiple applications from a single terminal Session
  - To VTAM applications on z/OS or OS/390
  - To TCP/IP applications
- Supports access from terminal sessions in both SNA and TCP/IP networks
- 3270 terminals, 3270 emulators, and Telnet TN3270 clients
- Runs on z/OS or OS/390, and normally runs as a VTAM application
- VTAM not needed if a TCP/IP-only environment
- Provides data compression between terminal and application
- Supports External Security Manager for user authentication
- Standard SAF interface supported
- Product number: 5655-K01

Figure 16-18. IBM Session Manager CI207.0

Notes:

IBM Session Manager for z/OS V1.1 is a session manager for VTAM and TCP/IP which provides:

- A secure and user-friendly method of accessing multiple OS/390 or z/OS systems from a single 3270 terminal session
- A single password-protected menu from which users can access all applications running on any z/OS or OS/390 machine in the network
- Logoff procedures, security checking, audit logging, and centralized administration, operations, and monitoring

IBM Session Manager Overview

- Provides a common user interface for access to VTAM and TCP/IP applications
- Password-protected application selection menu
- Single keystroke or simple command to connect to applications or switch sessions
- Windowing support to enable viewing of multiple applications at same time
- Cut and Paste facility to allow data to be copied between application sessions
- Provides an extensive Panel and Script Language (TPSL)
- Enables automation of application input/output sequences
- Enables screen panels and menus to be customized

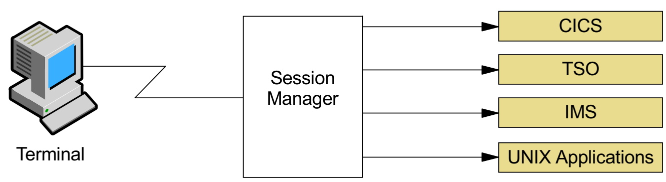


Figure 16-19. IBM Session Manager Overview CI207.0

Notes:

Users can switch between sessions with ease, or from the menu the user can point and click to create a new session to another VTAM or TCP/IP application.

Session Manager includes all the tools and facilities required to provide administration, security, systems, and performance management, as well as problem determination. It provides many features and facilities which enable users to improve their productivity, including:

- Only a single keystroke or simple command is required from a menu panel to connect a user to a critical application.
- Multiple applications can be viewed at the same time using the Sessions Manager's Windows Facility.
- The Cut and Paste facility allows information to be copied between the different application views.
- Using the Panel and Script Language as a preprocess input; user commands can also have alternative forms.
- The Record-Replay facility gives users the ability to capture session details for education, or perhaps to be viewed later as part of a problem determination effort.

16.7 CBEP

CBEP

- **CICS Business Event Publisher for MQSeries**
  - **Creates and queues MQSeries messages based on events occurring within existing CICS applications**
  - **Product function**
  - **Monitors events emanating from CICS applications**
  - **Based on user-specified selection criteria**
  - **Creates messages from associated data**
  - **Based on a user-specified message content rules**
  - **MQPUTS messages to MQSeries queue**
  - **User-specified**
- **Possible uses:**
  - **Extending existing CICS application**
  - **Application integration**
  - **Data source for message brokers**
- **Product number: 5655-J99**

Figure 16-20. CBEP CI207.0

Notes:

CICS Business Event Publisher for MQSeries (CBEP) is a powerful tool that gives you the ability to enhance and extend the capabilities of legacy and new applications running in CICS TS. It accomplishes this by allowing you to define rules that cause the CICS activities associated with business events to be monitored and potentially generate messages destined for MQSeries message queues. CBEP performs its tasks transparently to the CICS application program. No code changes are required in order to take advantage of the features offered by CBEP.

Potential uses for CBEP:

- Enterprise Application Integration (EAI). By giving you the ability to “connect” existing legacy applications to each other or to new applications, CBEP gives you a powerful tool to integrate your company's information processing. Applications can be integrated by using CBEP's ability to publish business events to other applications within your organization.
- Business to Business (B2B) integration. CBEP greatly simplifies the effort needed to drive the processes of business partners from your legacy applications. Often, the

information you need to make available to a business partner lives deep inside an existing application.

- Web enablement. CBEP lets you further extend web-enabled applications by publishing the events and data associated with them.
- Message broker front-end. Several message broker products are available for MQSeries. Brokers can provide tremendous flexibility for message alteration and distribution. However, all depend on the original creation of the message being performed outside the product. CBEP fills the gap between legacy applications and message brokers by providing an easy way to generate messages out of legacy applications that can be directed to a broker's message queues for further manipulation and distribution.

CBEP Major Components

- Message server
- Puts messages to a queue
- Event source connectors
- Builds messages on selective events
- Data space server
- Holds in memory rules database
- Rules database and engine
- VSAM storage for rules
- Workstation administration client
- Create and maintain rules database objects

Figure 16-21. CBEP Major Components CI207.0

Notes:

The CBEP architecture consists of the above objects. The following pages will provide additional detail.

Message Server

- Responsible for putting message on queue
- Ensures transactional unit of work integrity
- Multitasking
- Multiple message servers per z/OS image
- Load balancing
- Test versus production
- Multiple connections per message server
- Operator console support

Figure 16-22. Message Server CI207.0

Notes:

The CBEP message server operates as a long-running started task or batch job. Its primary function is to accept event messages from event source connectors and to write the messages to the appropriate MQSeries message queues.

The **operator interface** accepts and executes MVS Modify commands originating from the operator console.

The **set** component consists of one or more subtasks defined by the user to manage one or more event sources. The subtasks run in the message server address space. Sets are defined in the CBMPARMS library in the MSETS member. Each set that you define results in an instance of a set subtask. Set subtasks are responsible for allocating storage areas to temporarily hold messages generated by event source connectors, attaching (and dispatching) the qtasks that ultimately write the messages to MQSeries queues, and monitoring the event sources connected to the set.

The **qtask** component consists of a number of MVS subtasks attached during message server initialization by each set subtask. The number of qtasks attached for each set is defined by the MQTASKS parameter in the MSETxxxx member in the CBMPARMS library.

Qtask subtasks are responsible for writing event messages to the appropriate MQSeries message queue.

Event Source Connector

- z/OS subsystem-dependent
- CICS TS
- Other (future)
- Monitors events within subsystem (CICS)
- Builds messages for selective events
- Calls user exits, if necessary
- Alter/suppress/accumulate message
- Change MQPUT options
- Handles message-to-message server

Figure 16-23. Event Source Connector CI207.0

Notes:

The CBEP event source connector is responsible for monitoring events and applying user-built rules to determine their eligibility for publishing. Version 1 of CBEP provides an event source connector for CICS. The following information describes the architecture and function of the CBEP CICS event source connector. The CICS event source connector consists of three components. These components work together to:

- Monitor events occurring within CICS applications
- Compare the properties of the events to user-defined rules
- Where appropriate, create a message from the data associated with the event and pass it to the CBEP message server to write to an MQSeries message queue

The CICS event source connector can do the following:

- Monitor VSAM file control, temporary storage, transient data, interval control, and program control (LINK) requests
- Use the CICS application response code from the event as a method to determine eligibility

- Accumulate multiple messages within a single unit of work into one message to reduce network overhead
- Allow a user-provided exit to exclude an event after the rule has determined that the event may be eligible for publishing
- Allow a user-written exit program to alter a message once it has been created but before it has been published

Data Space Server

- Storage for persistent data
- CICS transaction patterns
- Holds in memory rules database
- Supports multiple message servers

Figure 16-24. Data Space Server CI207.0

Notes:

The CBEP data space server is a long-running MVS started task or batch job. Its purpose is to provide a semi-persistent mechanism for maintaining the state of CBEP message servers and event source connectors across warm starts of the servers and connectors. A data space server must be available in every MVS image in which you want to run a message server. You may have multiple data space servers — such as a production data space server and a test data space server — in a single MVS image.

The data space server owns the MVS data spaces used by its connected message servers (and their connected event source connectors) that are critical to their ability to shut down and restart without losing valuable state information. An example of this state information is the pattern knowledge base CBEP connectors built during the processing of events. By maintaining this information in a data space owned by the data space server, you can shut down and restart your CICS region without losing the information.

Rules Database/Engine

- Resides on VSAM file
- Populated and maintained by Windows-based GUI
- May be shared by multiple message servers
- Dynamically refreshable
- Add/alter/delete rules
- Maintain unit-of-work integrity
- Rules consist of:
  – Event selection criteria
  – Message layout specifications
  – Destination queue and queue manager names
  – MQSeries MQPUT options

Figure 16-25. Rules Database/Engine CI207.0

Notes:

The CBEP rules database component is responsible for maintaining and administering user-defined rules used for the selection and processing of business events.

The rules database consists of a set of user-defined rules. The rules database is stored in a standard VSAM data set that acts as a permanent repository for the rules and their associated objects (rule groups and group lists). All activity between the rules database and the workstation administration client is done to the rules database file. That is, objects that are downloaded from or uploaded to the rules database from the workstation administration client are downloaded from or uploaded to the rules database file repository.

During the initialization processing of a CBEP message server, the contents of rule groups as specified by a group list (supplied by a parameter in the MMAIN parm member in CBMPARMS) are read from the rules database file and stored in an *in memory* rules database. The *in memory* rules database is located in a data space owned by the data space server to which the message server is connected. All event monitoring and selection by the event source connectors associated with the message server instance are based on the contents of the *in memory* rules database.

Workstation Administration Client

- **Used to create and maintain rules database objects**
- **Communicates with TCP/IP listener subtask in message server**
- **Lock feature prevents multiple users from updating the same object**

Figure 16-26. Workstation Administration Client CI207.0

Notes:

The Workstation Administration Client provides an intuitive, easy-to-use Windows-based solution for maintaining rules to be used by the message servers and event source connectors. Rules are arranged and maintained as elements of rule groups to facilitate easy administration and implementation of the rules. Likewise, group lists made up of rule group names are used to designate which rule groups (and therefore, which rules) are to be used by an instance of a message server in building its *in memory* rules database.

The Workstation Administration Client is capable of downloading objects from and uploading objects to the mainframe message server. Locks are maintained on the rules database repository file to prevent multiple users from accessing the same object simultaneously and possibly corrupting the object.

The CBEP TCP/IP listener provides the functions needed to communicate between the workstation administration client and the CBEP message server.

CBEP System

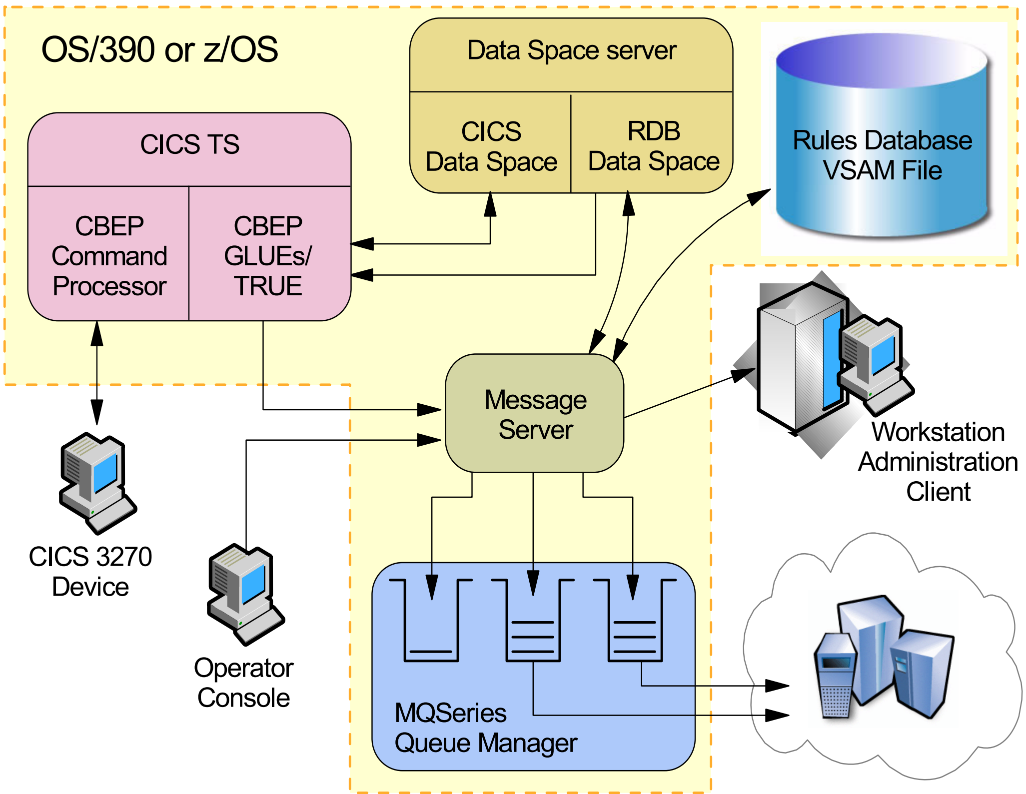


Figure 16-27. CBEP System CI207.0

Notes:

Software requirements:

- OS/390 V2R10 or above, or z/OS V1 R1 or above
- CICS Transaction Server V1R3 or CICS Transaction Server V2R2
- MQSeries for OS/390 V2.1 or above, or WebSphere MQ for OS/390, or WebSphere MQ for z/OS
- ISPF V4R5 or above, if using the CBEP COBOL copybook conversion utility
- TCP/IP for OS/390 V2R6 or above, for communicating with the CBEP workstation

Workstation requirements:

- Intel-based PC (or compatible) with minimum Pentium III processor and 256 MB RAM
- 20 MB available disk storage
- Windows 2000 or Windows XP operating system
- TCP/IP connectivity to mainframe computers running CBEP

Checkpoint Questions

1. **True or False? All CICS tools are included without charge with CICS TS V2.3.**
2. **True or False? CICS PA uses CICS, DB2, and WebSphere MQSeries accounting data.**
3. **True or False? CICS OTTO will optimize TCP/IP data streams.**
4. **True or False? CICS IA stores reports in a DB2 database.**
5. **True or False? The CICSVR address space is required to perform a forward recovery.**
6. **True or False? IBM Session Manager can connect to UNIX applications running under z/OS.**

Figure 16-28. Checkpoint Questions CI207.0

Notes:

Write down your answers below:

1.

2.

3.

4.

5.

6.

IBM Program Products: Summary

The IBM Program Products provide:

- Discovery
  - CICS Interdependencies Analyzer
- Development
  - CICS Business Event Publisher for MQSeries
- Deployment
  - CICS Performance Analyzer
  - CICS Online Transmission Time Optimizer
  - IBM Session Manager
  - CICS VSAM Recovery

Figure 16-29. IBM Program Products: Summary CI207.0

Notes:

Unit Summary

Having completed this unit, you should be able to describe the CICS tools that help you to manage and control your system:

- IBM Performance Analyzer for z/OS
- IBM CICS Online Transmission Time Optimizer
- IBM CICS Interdependency Analyzer
- CICS VSAM Recovery
- IBM Session Manager
- CICS Business Event Publisher for MQSeries

Figure 16-30. Unit Summary CI207.0

Notes:

Unit 17. Sysplex Support Functions (Optional Unit)

What This Unit Is About

This unit introduces these CICS TS functions that are particularly designed to be used within a z/OS sysplex.

The basic idea of these sysplex-related functions is the principle of shared data which is physically stored in a coupling facility, and thus is accessible from all active CICS regions in the sysplex.

This results in increased — actually 100% — availability of data and applications, and provides the basis for a flexible and intelligent workload management between systems and regions.

In the CICS TS context, this embraces the following options:

- VSAM record level sharing
- Shared data tables
- Shared temporary storage

What You Should Be Able to Do

After completing this unit, you should be able to:

- Describe the VSAM record level sharing function, with its special states and requirements, and define VSAM RLS files both to ICF and to CICS
- Describe the shared data table option and state its benefits
- Describe the concept and name the components of shared temporary storage pools
- Define shared TS queues to CICS TS, using both the RDO method and the DFHTST macro

References

CICS Transaction Server for z/OS Version 3.1 Installation Guide, GC34-6426

CICS System Definition Guide, SC34-6428

CICS Resource Definition Guide, SC34-6430

CICS Shared Data Tables Guide, SC34-6250

Unit Objectives

After completing this unit, you should be able to:

- Describe the VSAM Record Level Sharing function, with its special states and requirements, and define VSAM RLS files both to ICF and to CICS
- Describe the Shared Data Table option and state its benefits
- Describe the concept and name the components of Shared Temporary Storage Pools
- Define shared TS queues to CICS TS, using both the RDO method and the DFHTST macro

Figure 17-1. Unit Objectives CI207.0

Notes:

17.1 VSAM Record Level Sharing

What Is VSAM RLS?

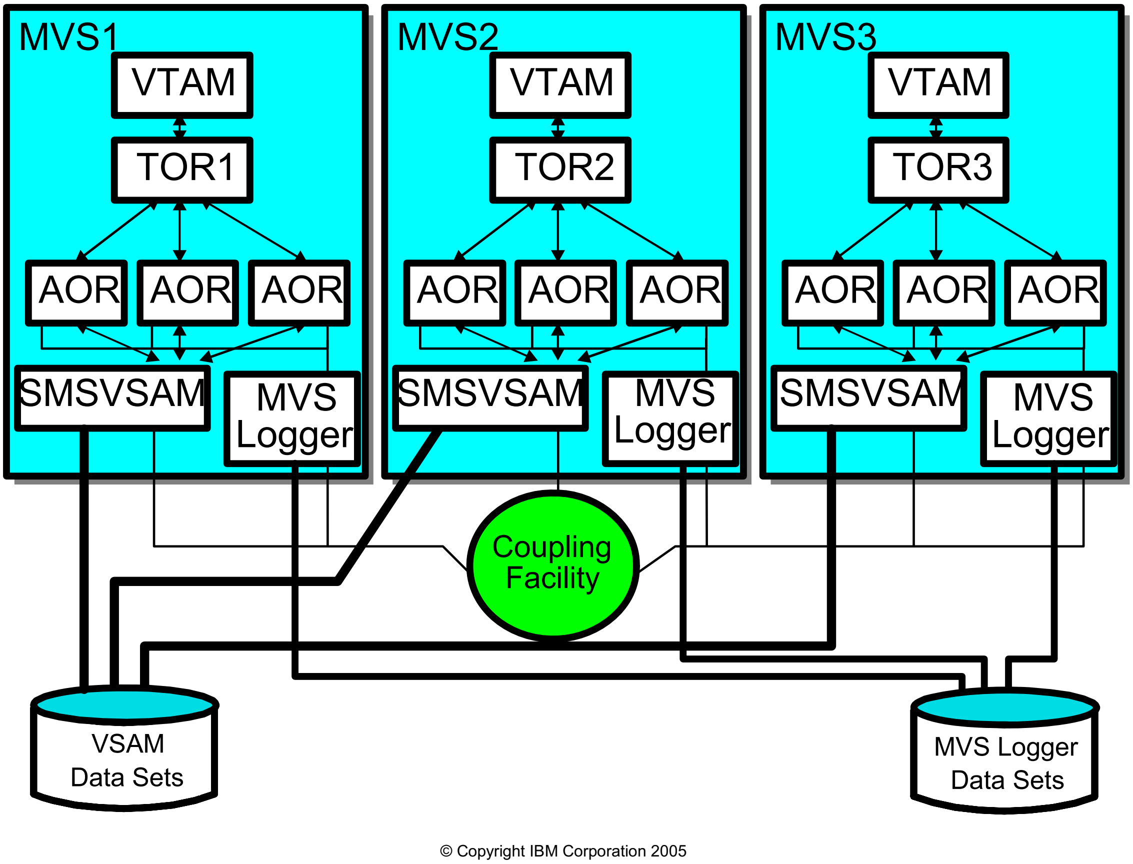


Figure 17-2. What Is VSAM RLS? CI207.0

Notes:

- Remember, in non-RLS mode, only one CICS region can open a VSAM data set for update. Any other regions that need access to these files have to function ship their requests to the file owning region.
  - The shared data tables function is restricted to one MVS image, and is advantageous for READ operations only.
- VSAM RLS allows multiple CICS regions to open the same data set concurrently with full update capability.
  - A parallel sysplex with a coupling facility is required.
  - The CICS regions must all run in the same parallel sysplex.
  - There must be one SMSVSAM server started in each MVS image.
  - DFSMStvs adds the ability to share VSAM data between CICS and batch for update as well as for reads.
  - User data files must be SMS-managed, and must not use the IMBED attribute.
- VSAM control blocks and buffers are held in a data space owned by SMSVSAM.

VSAM Recovery Attributes and Disposition

DEFINE CLUSTER attributes **held in the VSAM ICF catalog**

- **LOG(UNDEFINED | NONE | UNDO | ALL)**
  Type of recovery required
- **LOGSTREAM(logstream_name)**
  Forward recovery logstream name
- **BWO(UNDEFINED | TYPECICS | TYPEIMS | TYPEOTHER)**
  Backup-while-open eligibility

Changed effect of the JCL DISP= specification

- **DISP= SHR | OLD**
  Sharing is between address spaces

Figure 17-3. VSAM Recovery Attributes and Disposition CI207.0

Notes:

- **UNDEFINED** attributes may be defined in the CICS file resource definition.
- The existence and availability of a **log stream** is not checked before CICS TS attempts to open the data set in RLS mode.
- The DISPosition JCL specification has a different and extended function for RLS data sets, as VSAM shareoptions are ignored in RLS access mode.
- Recoverable data sets may only be opened for READ by batch jobs, when opened by CICS in RLS mode.
- Nonrecoverable data sets may be opened for READ/WRITE both by CICS regions and by batch jobs concurrently.

DISP=OLD	• First open establishes exclusivity. • More OPENs and CLOSEs may be issued by the same address space. • No other address space can issue OPEN or CLOSE.
DISP=SHR	• Multiple concurrent WRITEs and READs are assumed.

Level of READ Integrity

Resource Definition

- **DEF FILE(name)**
  `RLSACCESS(YES)`
  ....

Application Programming

- **EXEC CICS READ**
- **EXEC CICS READNEXT**
- **EXEC CICS READPREV**

READINTEG(UNCOMMITTED)
"Dirty read" of current VSAM record without integrity.

READINTEG(CONSISTENT)
Only committed data is returned.

READINTEG(REPEATABLE)
Only committed data is returned. No update allowed by other tasks until reading task commits.

Figure 17-4. Level of READ Integrity CI207.0

Notes:

- Without VSAM RLS, there is no READ integrity for shared file access. Data may be returned to an application that is backed out subsequently.
- The `READINTEG` option allows it to be specified that an application may be returned committed data only.
- If the record requested is in the process of update by another task, the reading task has to wait:
  - For a recoverable file, until the updating task completes
  - For a nonrecoverable file, until the VSAM request of the update process (ADD, WRITE, REWRITE) completes.
- `READINTEG` does not apply to files accessed in non-RLS mode.
- If the application program specifies a `READINTEG` option, this will override any file resource definition-supplied option.

Active and Retained Locks

- Locks may be EXCLUSIVE or SHARED.
- Locks are ACTIVE when first acquired; may later become **RETAINED**.
- Locks are allocated to the CICS region and the unit of work under which the file access occurs, ***CICS-applid.UOW-id*** being registered as the lock owner name.
- Depending on the file's recovery attributes, the READINTEG option, or both, locks are released when the VSAM access has completed or when the unit of work completes.
- In case of failures, active locks are converted to RETAINED
  - At CICS's request, when UOW fails
  - At SMSVSAM's request, when CICS fails
- Locks are maintained in the IGWLOCK00 CF structure.

Figure 17-5. Active and Retained Locks CI207.0

Notes:

- **EXCLUSIVE** locks are used to protect update to data sets, both recoverable and nonrecoverable.
- **SHARED** locks are used to support read integrity. They ensure that a record is not in the process of being updated during a read request or a series of read requests; in other words, no other task or UOW may establish an EXCLUSIVE lock as long as one or more shared locks are active for a certain record.
- Locks are held on an update to a nonrecoverable data set for the duration of the request, for example from `READ UPDATE` to `REWRITE`.
- Locks on recoverable data sets are held until the end of the UoW, for example from `READ UPDATE` to `SYNCPOINT` or end of task.
- **RETAINED** locks ensure that data integrity is maintained for records that were owned by failed units of work until the unit of work can complete, normally through a backout process.

VSAM RLS Components

- SMSVSAM provides:
  - Server code
  - Buffer management
- CICS TS provides:
  - Unit of work control through the recovery manager
  - Logging of before and after images data for backout and forward recovery using the MVS logger services
- Coupling Facility provides:
  - Locking (access control)
  - Cache (data buffering)

Figure 17-6. VSAM RLS Components CI207.0

Notes:

- VSAM now acts as a subsystem, in a similar way to DB2 and IMS.
- CICS issues the request for read/write, as well as the locking and logging requests.
- SMSVSAM performs the reading and updating, and the management of VSAM buffers.
- MVS system logger manages the actual log.

VSAM Access Modes

- Shared mode:
  - Record Level Sharing (RLS)
- Non-shared mode:
  - Non-shared resources (NSR)
  - Local shared resources (LSR)
- Mode is determined at FILE OPEN time
- All data sets within a sphere must use the same mode
  - Usually are defined with SHAREOPTION(2,3)
- Mode can be switched only when:
  - All files in sphere are closed
    and
  - There are no retained locks

Figure 17-7. VSAM Access Modes CI207.0

Notes:

- The mode of access is not a property of the data set itself; it is a property of the way the data set is opened:
  - A given data set can be opened in NSR mode at one time, and in RLS mode at another time.
- Multiple address spaces (in other words, CICS regions and batch jobs) can only access a data set concurrently using the same mode.
  - The first instance (CICS or batch) that opens a data set determines the mode in which it is accessed, presuming that the data set is enabled to be opened in any mode.
  - If, for example, a CICS region has opened a data set with RLSACCESS(YES), access is denied to any batch job or to another CICS region that attempts to open the same data set with RLSACCESS(NO).

RLS Quiesce - Switching Modes

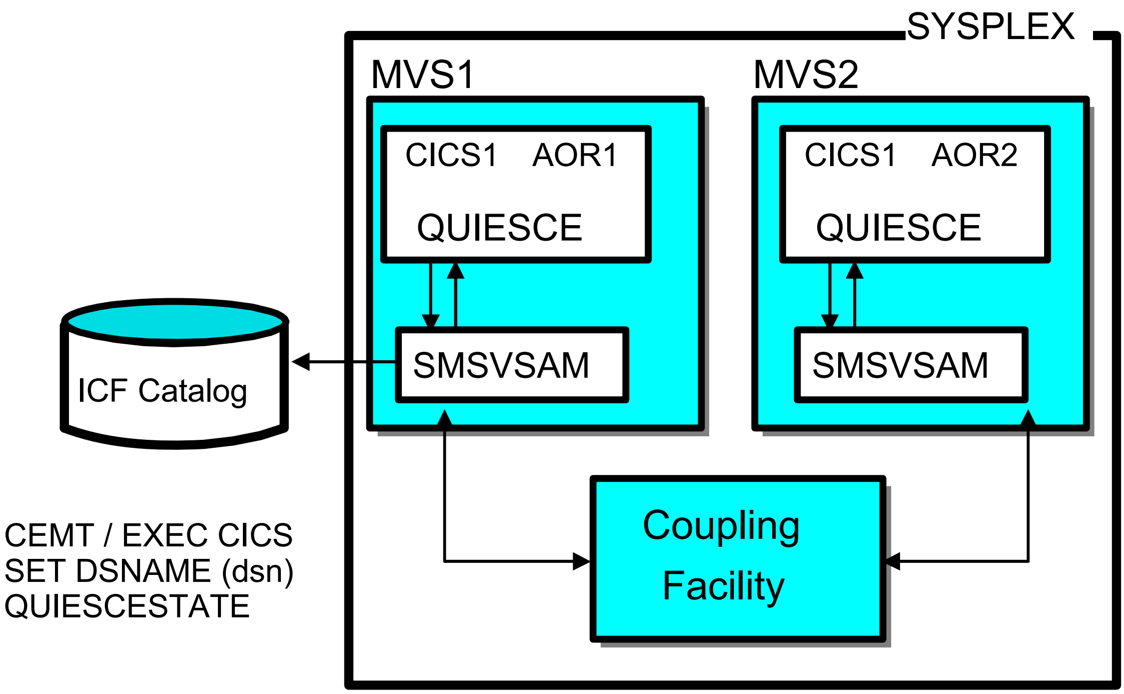


Figure 17-8. RLS Quiesce - Switching Modes CI207.0

Notes:

- **`SET DSNAME QUIESCESTATE`** is used to tell VSAM that RLS access to a file is to be discontinued.
  - The quiesce state flag in the data set's ICF catalog entry is turned ON, to reflect the fact that the data set can be opened in non-RLS mode only.
  - The command may be issued in any AOR, and then is propagated to all CICS regions across the sysplex that have access to the named data set.
- Command options for **`CEMT/EXEC CICS SET DSNAME(name) QUIESCESTATE ...`**

QUIESCED	Wait until all inflight UoWs complete.
IMMQUIESCED	Abend all inflight tasks and initiate quiescing.
UNQUIESCED	Remove flag from ICF catalog. Data set will be ready to be opened in RLS mode as well as in non-RLS mode.

- SIT parameter **`QUIESTIM`** determines how much time the quiesce process may take. Default is 240 seconds (4 minutes). If not completed in this time, SMSVSAM will cancel the quiesce.

VSAM RLS Related SIT Parameters

▪ **RLS= <u>NO</u> | YES**	Whether CICS is to support RLS or not
▪ **RLSTOLSR= <u>NO</u> | YES**	Whether to include RLS files in calculating the sizing of LSR pools not explicitly defined
▪ **QUIESTIM= <u>240</u> | nnn**	Timeout value (sec) for data set quiesce processing
▪ **FTIMEOUT= <u>30</u> | nn**	Default timeout interval (sec) for file access in RLS mode
▪ **TBEXITS= (,,,,,) | (pgm1,pgm2,..)**	User exit programs to be activated for six particular exit points

Figure 17-9. VSAM RLS Related SIT Parameters CI207.0

Notes:

- As deadlocks may now occur between different systems, the **FTIMEOUT** value has been introduced to indicate how long VSAM should wait before terminating a request and returning an exception condition.

  If the DTIMEOUT value is specified for a transaction definition, this will be used as the file timeout value for that transaction's VSAM requests.

- Refer to the *CICS Customization Guide, SC34-6429*, for, respectively, changed and new user exits for customizing transaction backout processing.

VSAM RLS RDO and API/SPI Options

- FILE resource definition

RLSACCESS= NO | YES	Whether CICS is to open a file in RLS mode or not
READINTEG=	The default level of READ integrity - only applicable to files opened in RLS mode

- Effects to API and SPI
  - **NOSUSPEND** option available to all file control commands
  - **LOCKED** and **RECORDBUSY** responses
  - READ integrity options to READ commands
  - Appropriate options on **EXEC CICS INQUIRE** and **SET DSNAME**

Figure 17-10. VSAM RLS RDO and API/SPI Options CI207.0

Notes:

- Remember that the FILE resource definition parameters, `RECOVERY`, `FWDRECOVERYLOG`, and `BACKUPTYPE`, are only meaningful for non-RLS files. For RLS files, the appropriate parameters have to be specified with the data set definition in the ICF catalog.
- The new parameters may also be specified by the CEMT or EXEC CICS INQ and SET commands.
- Application programs may have to be changed in order to handle the new exception conditions on file control command, but only if they either use the new `NOSUSPEND` option, or if READ integrity is not set to `UNCOMMITTED`.
- Refer to the *CICS System Programming Reference, SC34-6435*, for a complete list of parameters provided for the INQ and SET DSNAME commands.

Authorizing Access to an SMSVSAM Server

//.... JOB **USER=CIMAIN**
SIT: **APPLID=CICSAGT1**

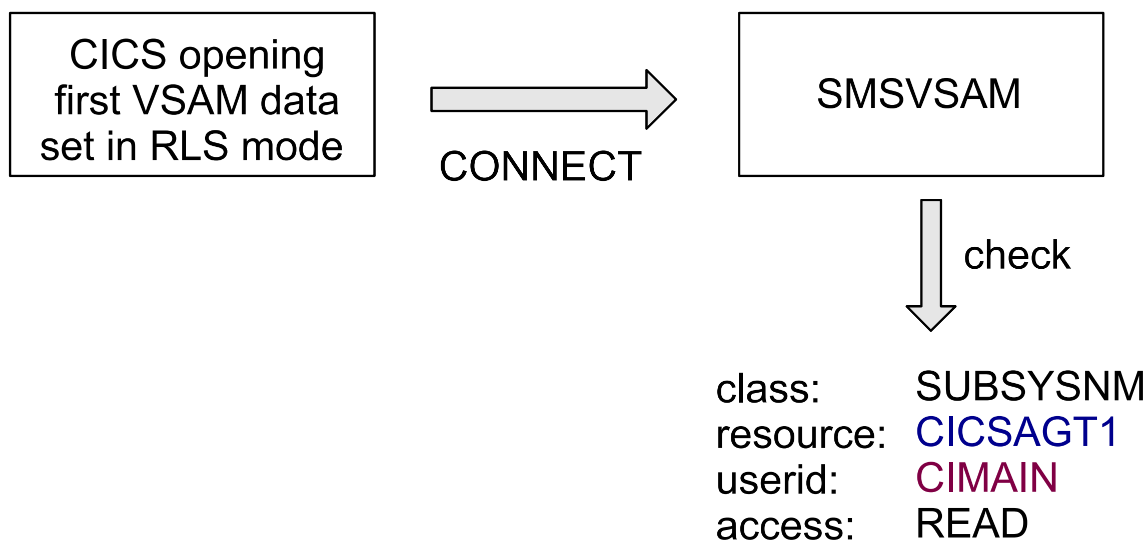


Figure 17-11. Authorizing Access to an SMSVSAM Server CI207.0

Notes:

- Protection ensures that CICS regions can only connect to SMSVSAM if the combination of the CICS TS's region userid and applid match the profile names defined to RACF.
- Profiles are in RACF class **SUBSYSNM**.
- The resource name is the **APPLID** defined in the SIT.
- The userid checked is the CICS Region Userid which requires **READ** permission for the protected resource.
- These are the RACF commands that define a profile and a permission in this area:

```
PERMIT SUSSYSNM applid UACC(none)
PERMIT applid CLASS(SUBSYSNM) ID(region_userid)
```

- The SUBSYSNM class must be active and RACLISTed for this protection to be in effect.

17.2 Coupling Facility Data Tables (CFDT)

Coupling Facility Data Tables (CFDT)

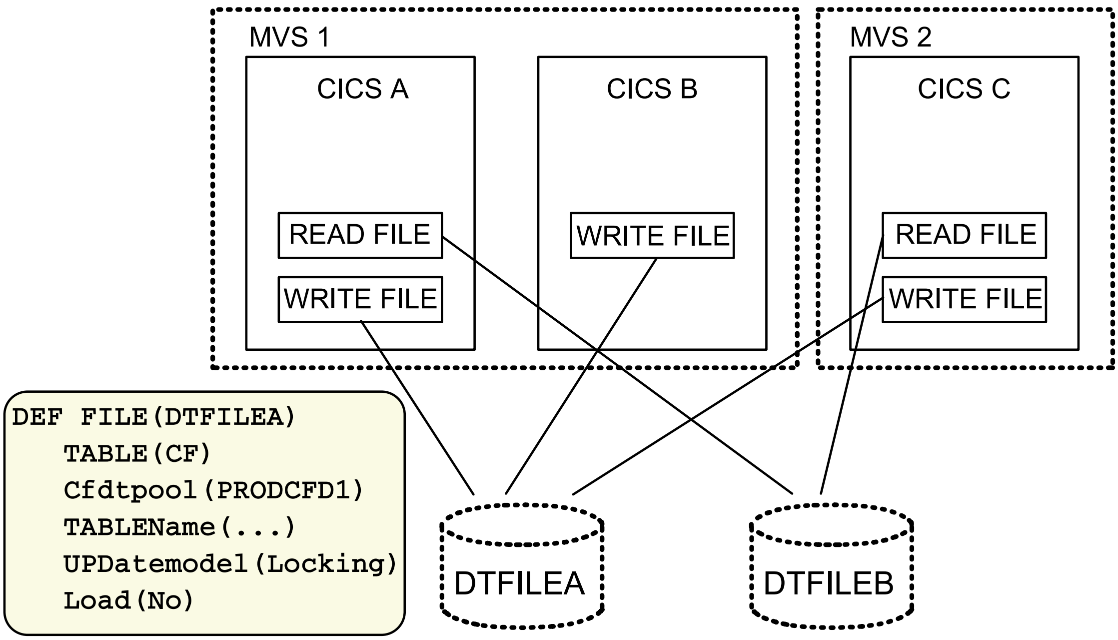


Figure 17-12. Coupling Facility Data Tables (CFDT) CI207.0

Notes:

- CICS TS extended the scope of sharing data tables to the sysplex, allowing for sysplex-wide sharing of data.
- Files that are to use this feature have to be defined as **coupling facility data tables (CFDT)**, as data is stored in **data table pools** of a coupling facility (see next foil).
- A specific **table name** may be specified. If omitted, it defaults to the file name.
- The `LOAD` parameter specifies whether the CFDT is to be loaded from a source data set when first opened.
  Unlike for CMT and UMT, there is no need for a source data set. But if there is one, it is handled the same way as for UMT, in other words updates to a CFDT are not reflected in the source VSAM data set.
- **Update integrity** of CFDT data is controlled by the `UPDATEMODEL` parameter.
  - `UPD(Contention)` allows for concurrent updates of records by multiple regions. This means that data read for update is not locked, and may be changed by another

application before the first one issues its rewrite. This is applicable to nonrecoverable tables only.

- `UPD(Locking)` implements locking (enqueue) mechanisms that ensure isolation of changes, as with file or UMT processing.
  For files (tables) defined as recoverable, record locks are held until the end of the unit of work, and backout is performed in the event of task failures, CICS region failures, and even MVS failures. *However, there is no forward recovery support. This means that, in the event of a CF structure failure, the data is lost.*

- CFDT data tables therefore may be particularly useful for so-called informal data, which is characterized as:
  - Data that is relatively short-term in nature
  - Data volumes that are not usually very large
  - Data that needs to be accessed fast
  - Data that commonly requires update integrity
  - Data whose occasional (though unlikely) loss can be tolerated by user applications.

CFDT Pools and Pool Servers

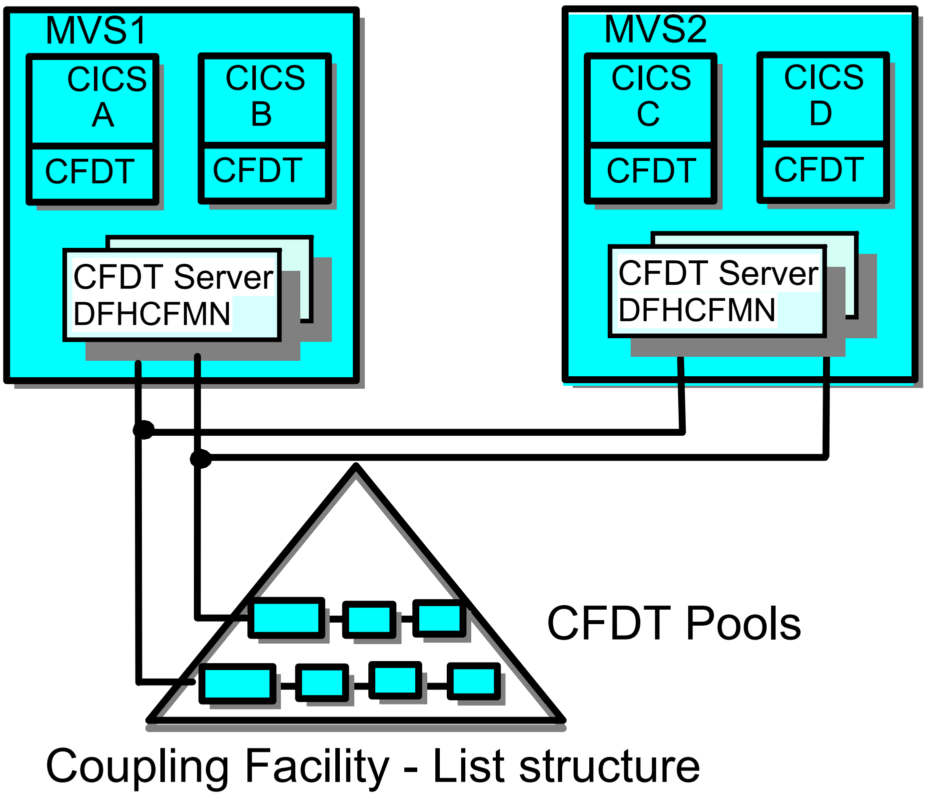


Figure 17-13. CFDT Pools and Pool Servers CI207.0

Notes:

- Coupling Facility Data Tables to be shared are stored in **CFDT pools**.
- Each CFDT pool corresponds to a coupling facility list structure that has to be defined in the coupling facility resource management (CFRM) policy. The structure name is fixed to the CFDT pool name, prefixed by the string `DFHCFLS_`.
  - Therefore, use of CFDT requires either a stand-alone or a LPAR-based coupling facility.
- Access to a CFDT pool is through a **CF data table server** address space.
  - The CICS-supplied CFDT Server program is named **DFHCFMN**.
  - Each CFDT Server provides access to *one* named pool.
  - A CFDT is allocated to a pool by means of the `CFDTPOOL` file definition parameter.
  - At least one CFDT Server is needed in each MVS image in the sysplex.
  - If multiple CFDT pools are defined, a CICS region may access more than one CFDT Server at a time.
  - Access to CFDTb Server is provided by Authorized Cross-Memory services (AXM) that must be defined in SYS1.PROCLIB(IEFSSNxx):

```
SUBSYS SUBNAME(AXM) INITRTN(AXMSI)
```

CFDT Server Startup JCL

```
//CICSCFDT  JOB ,...........
//*
//*          Start CICS CF Data Tables Server
//CFDTSTVR  EXEC   PGM=DFHCFMN,REGION=64M
//*
//*          authorized lib. containing DFHCFMN
//STEPLIB   DD   DSN=CICSTS.CICS.SDFHAUTH,DISP=SHR
//SYSPRINT  DD   SYSOUT=*
//SYSIN     DD   *
  FUNCTION=SERVER
  POOLNAME=PRODCFD1
  MAXTABLES=100       default value
/*
```


Figure 17-14. CFDT Server Startup JCL CI207.0

Notes:

- The `POOLNAME` parameter is required; there are other parameters to enable you to control, for example:
  - Server security
  - Server statistics
  - List structure attributes
  - Debug trace options
  - Lock wait intervals

  All parameters are described in the *CICS System Definition Guide, SC34-5988*.
- The same program (DFHCFMN) may be used to perform utility functions such as LOADing and UNLOADing shared data table pools, for example, in the event of a scheduled maintenance of the Coupling Facility; therefore a **FUNCTION** has to be specified.
- The CF list structure that is to hold the shared TS queue data must be named **DFHCFLS_poolname**, when defined using CFRM policy statements; the **POOLNAME** specified here must match the second part of this name.

Managing CFDT Servers and Pools

- `D XCF,STR,STRNAME=DFHCFLS_`*`poolname`*
- `SETXCF ALTER` - increase pool size
- `DFHCFMN - UNLOAD and RELOAD` functions
  - Move pool to another coupling facility (CF)
  - Preserve pool during CF maintenance
- `F jobname SET (DT server parameters)`
- `F jobname DISPLAY or PRINT`
  - `CONNECTIONS` CICS applids connected to this server
  - `BUFSTATS` Queue index buffer pool statistics
  - `CFSTATS CF` I/O and response statistics
  - `POOLSTATS` Pool list structure usage statistics
  - `STGSTATS` Main storage allocation statistics
- `F jobname STOP`
- `F jobname CANCEL`

Figure 17-15. Managing CFDT Servers and Pools CI207.0

Notes:

- The MVS Coupling Facility-oriented command SETXCF may be used to increase a TS pool size up to the maximum specified in the CFRM policy at the time the structure was allocated.
- DFHCFMN supports LOADing and UNLOADing complete TS pool contents. The reloaded pool does not need to have the same name as the original pool.
- TS data sharing servers are managed by means of **MVS MODIFY** (in brief: “F”) commands against the job or started task name of the server address space.
  - Any (SYSIN) parameters, except those indicated as being for initialization only, may be changed using the **SET** command.
  - As the **DISPLAY** command returns its responses to the console, **PRINT** produces the same output on the print file (SYSPRINT DD).
  - TS Server may be **STOP**ped waiting for any active connections to terminate.

CANCEL terminates the server immediately; this is likely to cause CICS transaction abends.

17.3 Temporary Storage Data Sharing

TS Data Sharing Overview (1)

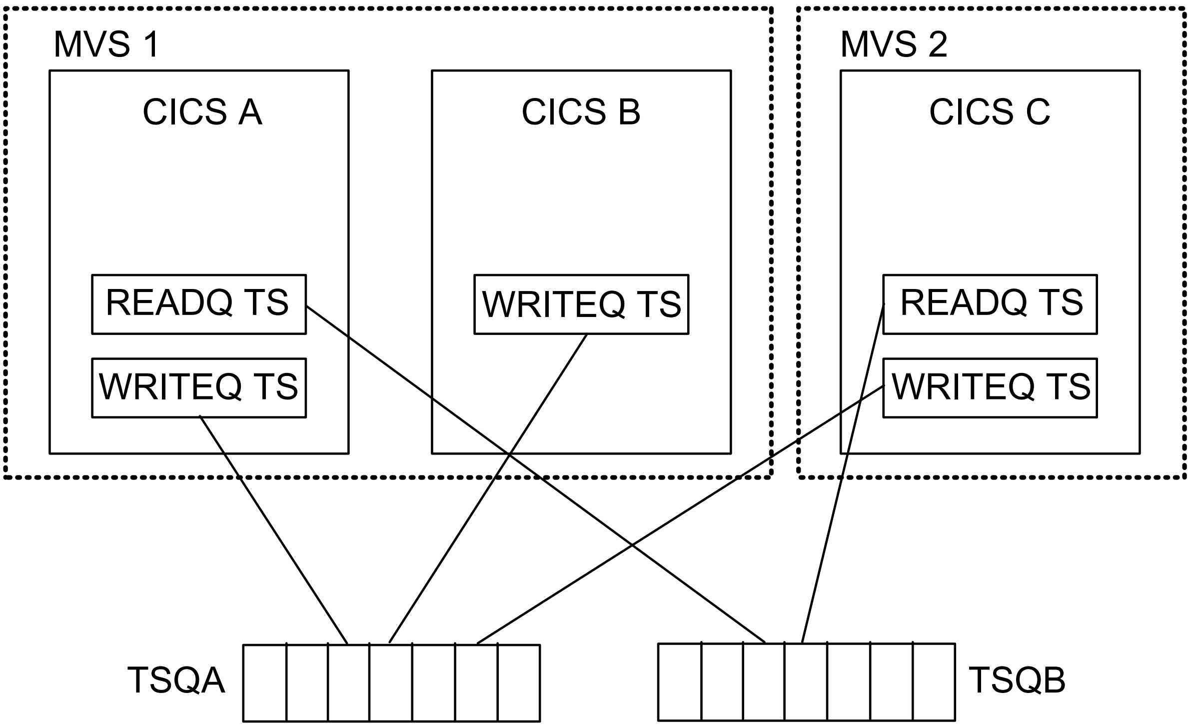


Figure 17-16. TS Data Sharing Overview (1) CI207.0

Notes:

- Temporary storage queues may be accessed by multiple CICS regions that are running on any MVS image within a **parallel sysplex with a coupling facility available**.
- Only nonrecoverable queues can be shared.
  - Unlike within a single region, these shared queues are normally preserved across a CICS restart, or even an MVS IPL, though nonrecoverable, as they are held in Coupling Facility storage which will normally be buffered for power failures.
- The benefits to applications:
  - Removal of dynamic routing/workload management affinities between transactions that communicate via TS queues
  - Simplified migration of existing CICS applications to the parallel sysplex environment
  - No more need to function ship TS requests to a Queue Owning Region

TS Data Sharing Overview (2)

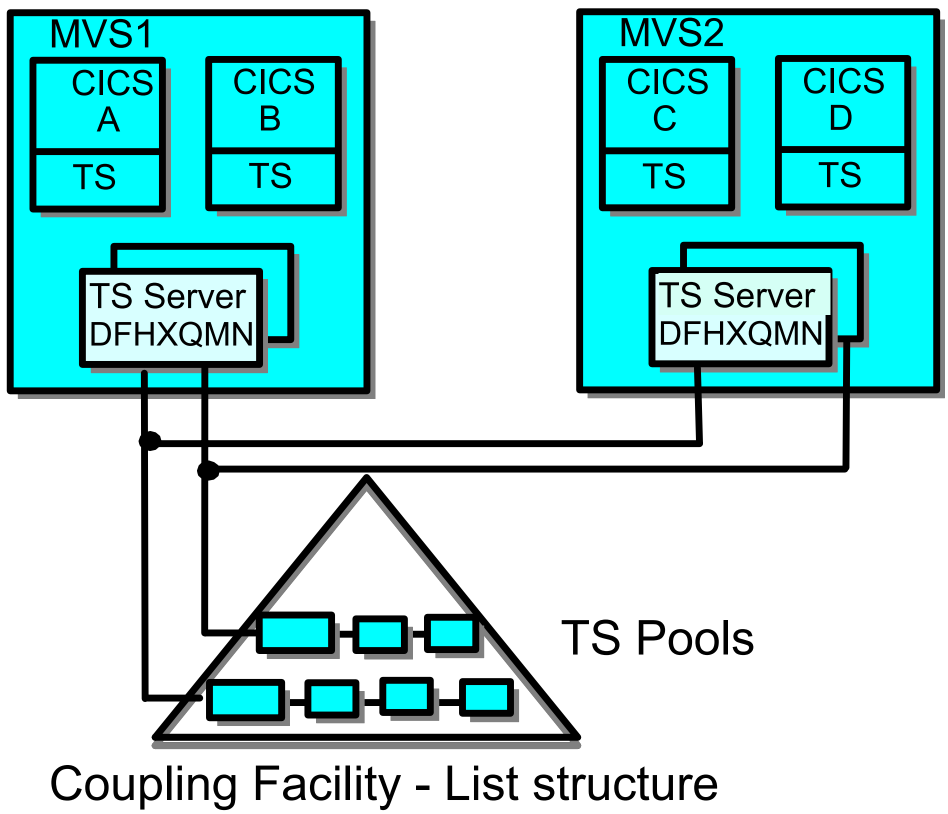


Figure 17-17. TS Data Sharing Overview (2) CI207.0

Notes:

- Temporary Storage queues to be shared are stored in **TS pools**.
- Each TS pool corresponds to a coupling facility list structure.
  - Therefore, use of TS Data Sharing requires either a stand-alone or an LPAR-based coupling facility.
- Access to a TS pool is through a **TS data sharing server** address space.
  - The CICS-supplied TS Server program is named **DFHXQMN**.
  - Each TS Server provides access to *one* named pool.
  - At least one TS Server is needed in each MVS image in the sysplex.
  - If multiple TS pools are defined, a CICS region may access more than one TS Server concurrently.
  - Access to TS Server is provided by Authorized Cross-Memory services (AXM) that must be defined in SYS1.PROCLIB(IEFSSNxx):

```
SUBSYS SUBNAME(AXM) INITRTN(AXMSI)
```

TS Pool Server Startup JCL

```
//CICSTSQS  JOB ,...........
//*
//*         Start CICS TS Data Sharing Server
//CFDTSTVR   EXEC    PGM=DFHXQMN,REGION=64M
//*
//*         authorized lib. containing DFHXQMN
//STEPLIB    DD   DSN=CICSTS.CICS.SDFHAUTH,DISP=SHR
//SYSPRINT   DD   SYSOUT=*
//SYSIN      DD   *
  FUNCTION=SERVER
  POOLNAME=TSHSHR1
  MAXQUEUES=1000     default value
  SMALLQUESIZE=32K   limit for "small Qs"
  POOLSIZE=0         default- no limit
  BUFFERS=100        default value
/*
```


Figure 17-18. TS Pool Server Startup JCL CI207.0

Notes:

- The same program (DFHXQMN) is used to perform utility functions such as LOADing and UNLOADing shared TS pools; therefore, a **FUNCTION** has to be specified.
- The CF list structure that is to hold the shared TS queue data must be named **DFHXQLS_poolname** when defined using CFRM policy statements; the **POOLNAME** specified here must match the second part of this name.
- **MAXQueues** specifies the maximum number of data lists to be reserved when the structure is allocated (will only be honored at list structure creation).
- The maximum amount of storage to be allocated for the list structure is specified by the **POOLSIZE** parameter. “0” means that the only the limit specified at CFRM policy definition will be applied.
- There are some more parameters which determine whether statistics data will be collected, at what rate, and so on.
  For a complete list of parameters, refer to the *CICS System Definition Guide, SC34-6428*.

RDO Definition for Shared TS Queues

CEDA DEFINE TSMODEL(name) GROUP(name)

TSmodel ==>
Group ==>
Description ==>

PRefix ==>

~~Location ==> Auxiliary | Main~~

~~RECovery ==> No | Yes~~

Security ==> No | Yes

POolname ==>

~~REMOTE ATTRIBUTES~~

Figure 17-19. RDO Definition for Shared TS Queues CI207.0

Notes:

Shared Temporary Storage queues have to be identified by a resource definition. There is no way to specify the shared nature of a queue directly from within the application.

- (Generic) TS queue names can be mapped directly to a shared TS pool by means of the TSMODEL resource definition, as shown in the visual.
  - Remember, the `PRefix` is the full or generic name of the TS queue, and may be up to 16 characters in length.
  - `POolname` specifies the 8-character name of the shared TS pool definition that you want to use with this TSMODEL definition.
- The `Location` and `Recovery` parameters are ignored for shared TS pool models.
- Likewise, no remote attributes can be set. If the `REMOTESystem` parameter is specified, the `POolname` parameter will be ignored.
- Multiple TSMODELs may point to the same TS pool.
- If a TS server cannot be connected, `SYSIDERR` is returned.

Shared TS Queue Macro Definition

```
DFHTST TYPE=REMOTE
       DATAID=FS,
       SYSIDNT=QOR1

DFHTST TYPE=REMOTE
       DATAID=REM,
       SYSIDNT=QOR2

DFHTST TYPE=SHARED
       SYSIDNT=SHR1,
       POOL=TSQSHR1

DFHTST TYPE=SHARED
       SYSIDNT=QOR2,
       POOL=TSQSHR2
```


Figure 17-20. Shared TS Queue Macro Definition CI207.0

Notes:

- The **TYPE=SHARED** option has been added to the DFHTST macro in order map remote sysids to a named Shared TS pool.
- Multiple `DFHTST TYPE=SHARED` entries for different SYSIDNTs may refer to the same TS pool.
- The main reason why this method of defining shared queues has been introduced and is still supported is that applications need not be changed when a QOR is migrated to Shared TS Queues, because:
  - If a remote queue is explicitly specified by an API command by use of the SYSID option, CICS will first map the SYSID to a matching SYSID entry on a `TYPE=SHARED` definition.
  - If an API command specifies a TS queue name for which a `TYPE=REMOTE TST` entry exists, CICS searches for a `TYPE=SHARED` entry with a matching SYSID.

  In both cases, CICS uses the `POOL` name to identify the TS server that manages the shared TS queue.

Managing TS Pools and Servers

- D XCF,STR,STRNAME=DFHXQMN_poolname
- SETXCF ALTER - increase pool size
- DFHXQMN - UNLOAD and RELOAD functions
  - Move pool to another coupling facility (CF)
  - Preserve pool during CF maintenance
- F jobname SET (TS server parameters)
- F jobname DISPLAY or PRINT
  - CONNECTIONS CICS applids connected to this server
  - BUFSTATS Queue index buffer pool statistics
  - CFSTATS CF I/O and response statistics
  - POOLSTATS Pool list structure usage statistics
  - STGSTATS Main storage allocation statistics
- F jobname STOP
- F jobname CANCEL

Figure 17-21. Managing TS Pools and Servers CI207.0

Notes:

- The MVS command SETXCF may be used to increase a TS pool size up to the maximum specified in the CFRM policy at the time the structure was allocated.
- DFHXQMN supports LOADing and UNLOADing complete TS pool contents. The reloaded pool does not need to have the same name as the original pool.
- TS data sharing servers are managed by means of **MVS MODIFY** (in brief: “F”) commands against the job or started task name of the server address space.
  - Any (SYSIN) parameters, except those indicated as being for initialization only, may be changed using the **SET** command.
  - As the **DISPLAY** command returns its responses to the console, **PRINT** produces the same output on the print file (SYSPRINT DD).
  - TS Server may be **STOP**ped waiting for any active connections to terminate.
  - **CANCEL** terminates the server immediately; this is likely to cause CICS transaction abends.

Authorizing Access to Shared Pool Servers

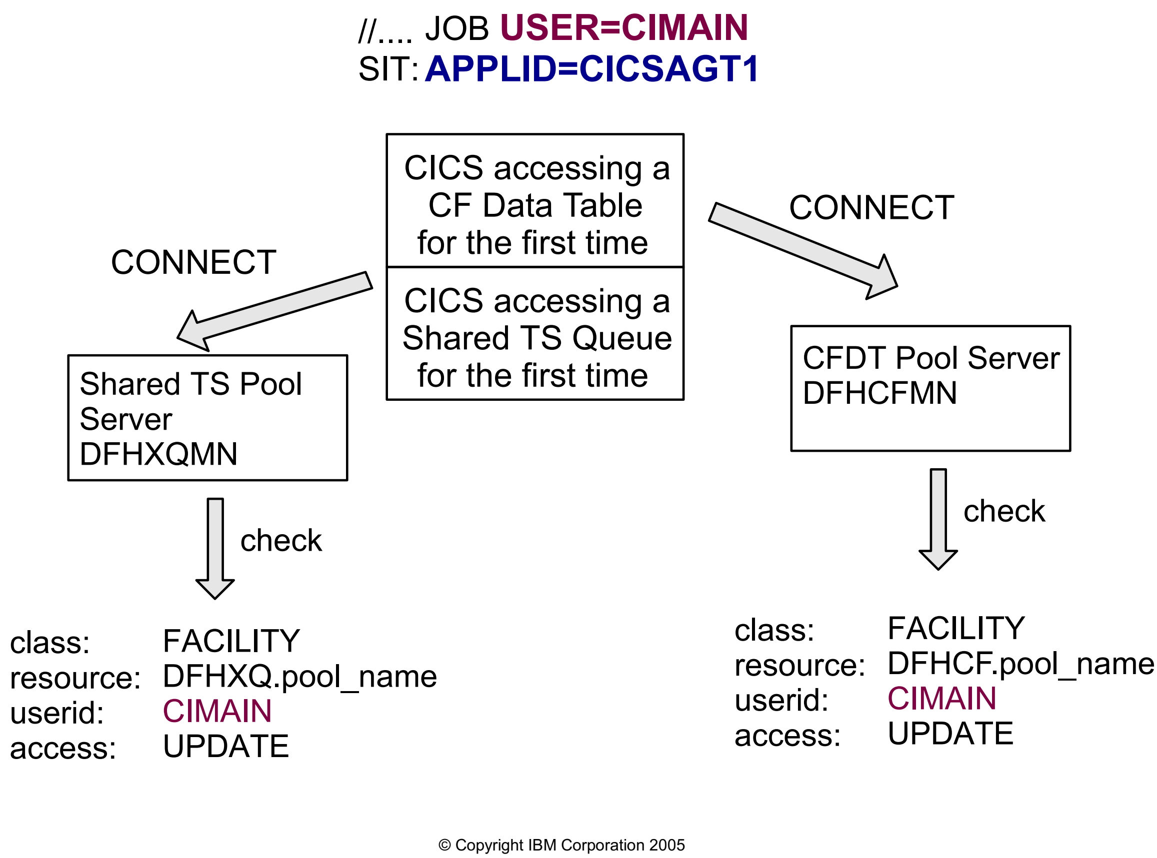


Figure 17-22. Authorizing Access to Shared Pool Servers CI207.0

Notes:

CICS TS's use of Shared Temporary Storage Servers and Coupling Facility Data Tables Servers can be protected in the same way as the connection to the SMSVSAM service.

- Profiles are in RACF class **FACILITY**.
- The resource name is the CFDT or Shared TSQ **Pool Name**, prefixed with the fixed string DFHXQ for a Shared TSQ Pool, and DFHCF for a CFDT Pool.
- The userid checked is the CICS Region Userid, which requires **UPDATE** permission for the protected resource.
- These are the RACF commands that define a profile and a permission in this area (the sample applies to a CFDT pool):

```
RDEFINE FACILITY DFHCF.pool_name UACC(none)
PERMIT DFHCF.pool_name CLASS(FACILITY) ID(region_userid)
```

Note that the access to the CF structures and to the pools by the appropriate servers is controlled by additional RACF profiles.

Sysplex Support Functions (Summary)

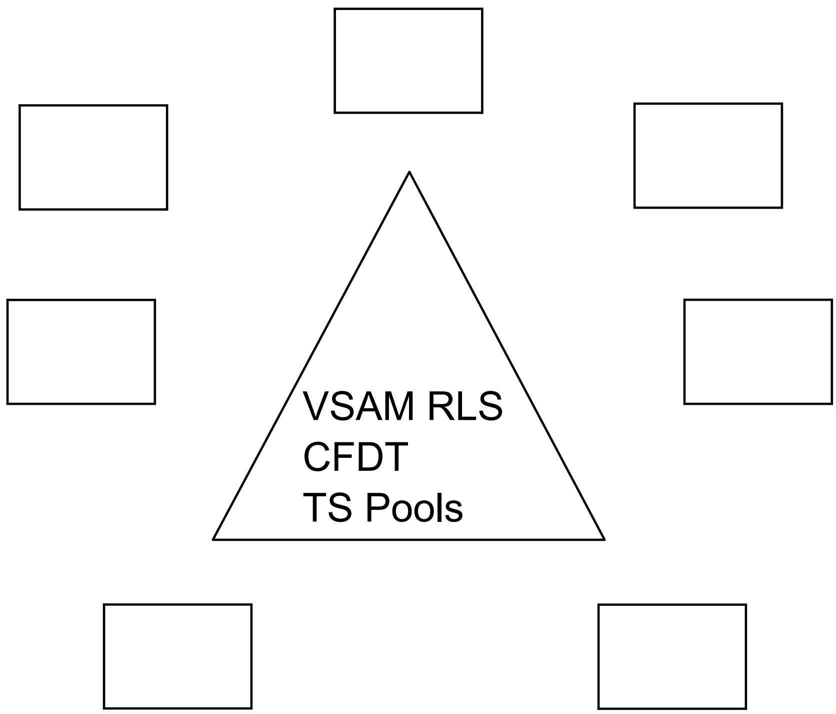


Figure 17-23. Sysplex Support Functions (Summary) CI207.0

Notes:

- High-availability, highly scalable, and high-performance application processing with shared data access within the scope of a z/OS parallel sysplex is supported by CICS TS for VSAM data, data tables, and temporary storage queues.
- VSAM data support is based on the VSAM Record Level Sharing (RLS) access method, where multiple CICS regions may connect to a VSAM server address space, which has to be there in every z/OS image and replaces the CICS file owning region.
- With the support of VSAM RLS, new integrity options have been introduced for file operations by CICS applications.
- The Coupling Facility Data Table function may be an interesting option to sysplex applications that handle so-called “informal data”.
- Shared access to nonrecoverable CICS Temporary Storage queues is provided by Shared TS pool servers that take the role of a queue-owning region.
  Shared TS queues are transparent to the application. SYSIDs may be mapped to TS pool server names by use of DFHTST entries.

Unit Summary

Having completed this unit, you should be able to:

- Describe the VSAM Record Level Sharing function, with its special states and requirements, and define VSAM RLS files both to ICF and to CICS
- Describe the Shared Data Table option and state its benefits
- Describe the concept and name the components of Shared Temporary Storage Pools
- Define shared TS queues to CICS TS, using both the RDO method and the DFHTST macro

Figure 17-24. Unit Summary CI207.0

Notes:

Unit 18. Accessing External Resource Managers' Data (Optional Unit)

What This Unit Is About

This unit introduces the CICS Resource Manager Interface (RMI) generally, and then shows at a high level the special facilities that support connection to DBCTL, DB2, and WebSphere MQ (formerly known as MQSeries).

What You Should Be Able to Do

After completing this unit, you should be able to:

- Describe in general the terms RMI, TRUE, and RM stub
- State what IBM resource manager systems are providing support for the CICS RMI
- Describe the system and application setup activities that are required to access:
  - DL/I data via Remote DL/I and DBCTL
  - SQL data via DB2
  - WebSphere MQ message queues
- List the restrictions that exist for the named system interconnection options

References

CICS Transaction Server for z/OS Version 3.1 Installation Guide, GC34-6426

CICS System Definition Guide, SC34-6428

CICS Resource Definition Guide, SC34-6430

CICS IMS Database Control Guide, SC34-6453

CICS DB2 Guide, SC34-6252

Unit Objectives

After completing this unit, you should be able to:

- Describe in general the terms RMI, TRUE, and RM stub
- State what IBM resource manager systems are providing support for the CICS RMI
- Describe the system and application setup activities that are required to access
  - DL/I data via Remote DL/I and DBCTL
  - SQL data via a DB2
  - WebSphere MQ message queues
- List the restrictions that exist for the named system interconnection options

Figure 18-1. Unit Objectives CI207.0

Notes:

18.1 Resource Manager Interface Concepts

External Resource Manager Access

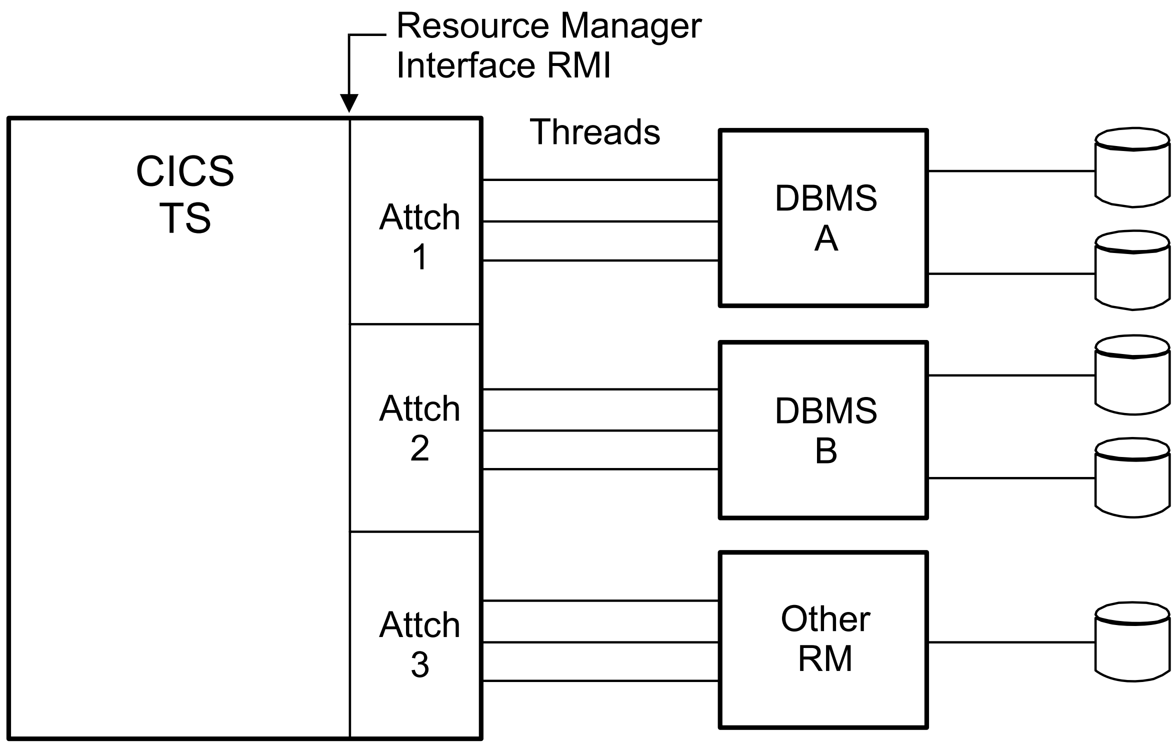


Figure 18-2. External Resource Manager Access CI207.0

Notes:

- CICS contains a general-purpose interface for communicating with non-CICS Resource Managers, such as Database Management Systems (DBMS), for granting CICS applications access to the external resources. This interface is called the **Resource Manager Interface** (RMI).
- For each External Resource Manager (ERM) connected, there must be an **Attachment** component specific to the ERM.
- The ERM attachment code operates in the CICS address space, and may be supplied either with CICS TS or by the ERM product.
- Attachments use MVS cross-memory services for communication with the ERM.
  - Logical connections between CICS and the ERM are called **threads**, and are normally implemented as separate MVS TCBs that operate in the CICS TS address space.
  - Thus the CICS TS region and the ERM system have to operate in the same MVS image.

Resource Manager Interface Environment

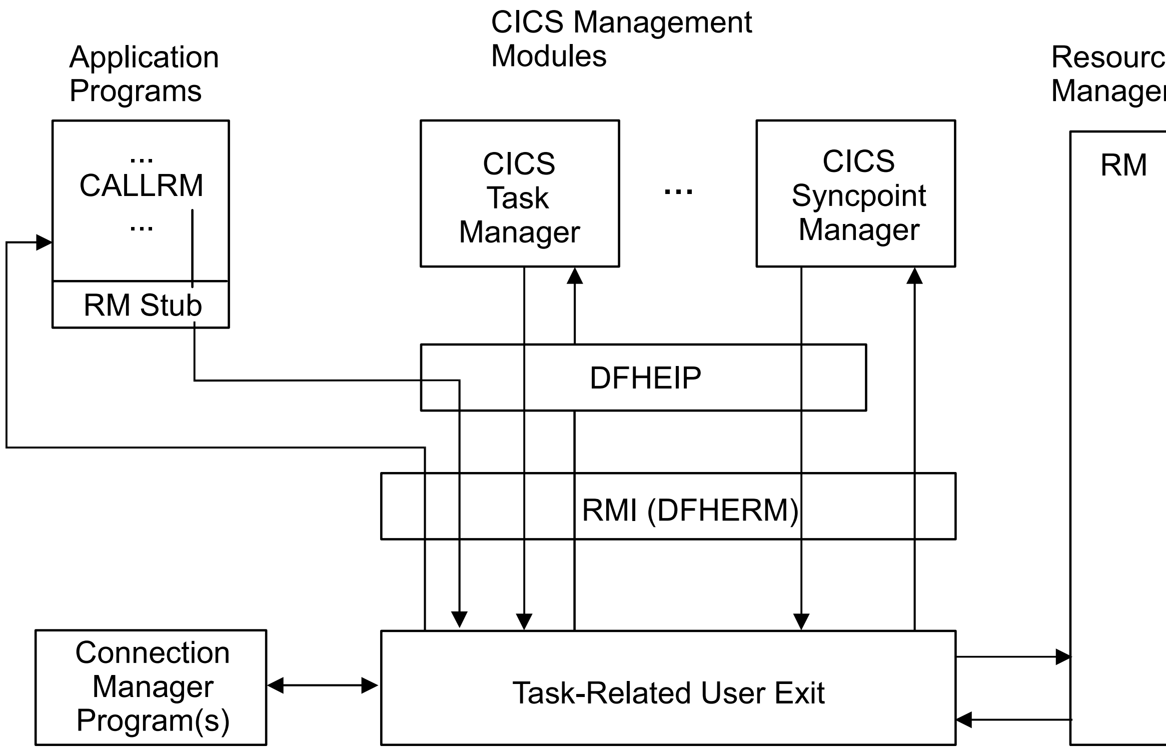


Figure 18-3. Resource Manager Interface Environment CI207.0

Notes:

- The ERM attachment code is implemented as CICS **Task Related User Exits** (TRUEs):
  - As opposed to global user exits (GLUEs), TRUEs action on behalf of single tasks and have access to task lifetime data.
  - TRUEs are activated by `EXEC CICS ENABLE` commands.
  - This `ENABLE` command is normally issued by a program that is supplied with the attachment, and typically is invoked through an entry in the CICS Program List Table (PLT) for initialization.
- Application programs use the original API of the resource manager whose data they want to access, for example **EXEC SQL** commands for DB2 data access.
  - These commands or calls have to be served by a program stub that is supplied by the appropriate attachment and has to be linked to the CICS application program.
  - This program stub calls the CICS EXEC interface program (DFHEIP) and names the TRUE component that has to be invoked via the RMI.

Syncpoint Coordination

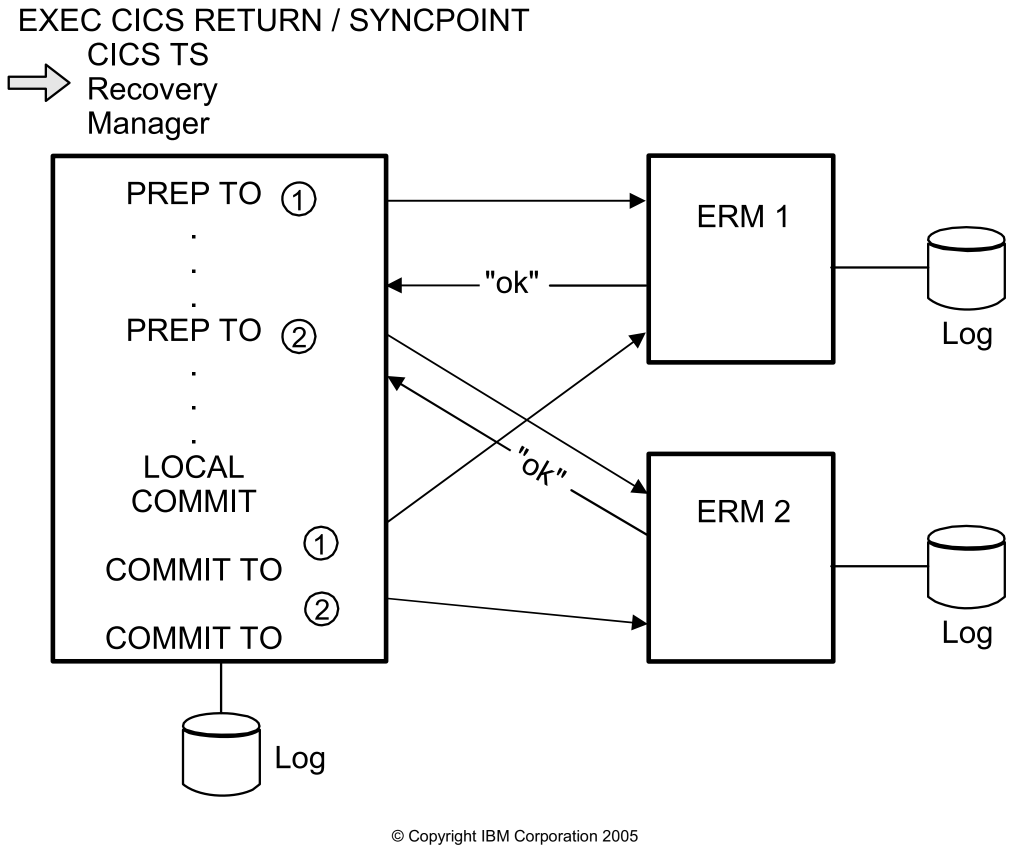


Figure 18-4. Syncpoint Coordination CI207.0

Notes:

- In addition to serving application requests, TRUEs will be called by CICS on task start, termination, or both, in order to process syncpoint coordination with the ERM, as specified by the `EXEC CICS ENABLE` command.
- CICS TS ensures that updates to ERM data performed by an application are handled as part of the CICS unit of work, just as updates to CICS resources are.
- On task termination, or when the application program issues an `EXEC CICS SYNCPOINT` command, CICS coordinates two-phase-commit syncpoint processing between all resource managers involved in the appropriate unit of work.
  - The CICS TS *Recovery Manager Domain* makes sure that all resource managers which are involved in the UoW are *prepared to commit* before they are requested to commit, and thereby release changed data.
  - CICS TS and the ERMs register each step of the syncpoint process on their log data sets in order to recover in a synchronized way if any of the systems or the connections between them fail during this process.

Typical System Setup Activities

- ✓ Activate DFHRMI resource definition group
- ✓ Request for automatic attachment connect:
  using either a PLT program or a SIT parameter, if available
- ✓ Define attachment related SIT parameters (if applicable)
- ✓ Define and activate attachment programs and transactions to CICS using RDO
- ✓ Define the External Resource Manager system to CICS
- ✓ Set up application related resource definitions (if applicable)
- ✓ Set up security for ERM access

Figure 18-5. Typical System Setup Activities CI207.0

Notes:

- The DFHRMI CICS-supplied RDO group is contained in DFHLIST.
- For the IBM resource manager products mentioned, automatic connection from CICS TS on startup is supported by use of CICS SIT parameters.
  The older method is to use an IBM-supplied PLT program, which is still supported.
- Attachments may be started and stopped manually by attachment-specific CICS transactions.
- At least the ERM's system name has to be defined to CICS in some way, which might be through SIT parameters or a CICS resource definition.
- Some attachments use SIT parameters to specify certain variable names or functions, for example the number of threads that may be concurrently active.
- For certain attachments, there are application-related specifications that have to be defined to CICS, so that the appropriate TRUE may access them.
- Setting up access authorizations for the ERM’s objects is a highly ERM-specific task, most of which is normally performed on the side of the ERMs.

18.2 Accessing DL/I Data

Accessing DL/I Data

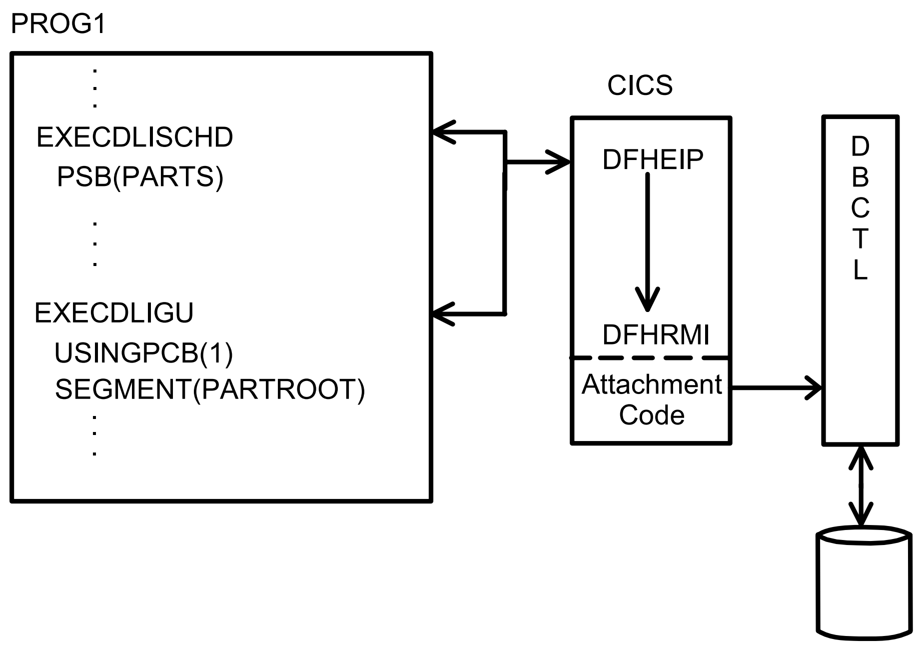


Figure 18-6. Accessing DL/I Data CI207.0

Notes:

- The ERM is either a DBCTL address space or a complete IMS environment with IMS Transaction Manager (TM) and IMS Database Manager (DM), which makes no difference from the CICS side.
- As in the native IMS/DBTCL environment, two application interfaces are supported for DL/I data access by CICS applications:
  - **EXEC DLI** commands
  - **CALL xxxTDLI** interface, using a parameter list.
- Before a program accesses a DL/I database, it must schedule the PSB describing the program's view of the data.
- Remember that a CICS TS region can only be connected to one DBCTL/IMS system at one time, and that both must operate in the same MVS image.
- CICS TS supports function shipping to CICS regions (Version 4.1 or older) that have the **Local DLI** function activated.

Application Considerations

`EXEC DLI`	XOPT(DLI) Translator option
`CALL xxxTDLI`	INCLUDE DFSLI00 on Link-Edit
`SDFHPROC(DFHEI*)`	CICS TS provided procedures for application preparation
`DBDGEN` `PSGEN` `ACBGEN`	Native IMS/DBCTL procedures

Figure 18-7. Application Considerations CI207.0

Notes:

- Application programming for CICS programs accessing DL/I data is the same as for native DL/I (IMS) programs, except that CICS applications have to use the `SCHD PSB` command or call to identify and schedule the PSB, and the `TERM` command, or call to terminate the PSB, respectively.
  - Only one PSB may be scheduled by an application program at a time, but after termination of a PSB, another may be scheduled by the same program.
  - As applications have to name the PSB name explicitly, there is no need to define allocations between transaction codes and PSBs at the system level.
  - Only **Remote PSBs** for DL/I function shipping have to be defined to CICS. DBCTL/IMS PSBs do not have to be defined.
- As the CICS translator handles EXEC DLI commands (with XOPT=DLI), there is no need for precompiling such programs.
- The PSB, DBD, and ACB control blocks come into effect in the DBCTL/IMS address space only, and therefore have to be defined and handled there, just as for native DL/I accesses.

CICS-IMS DBCTL Environment

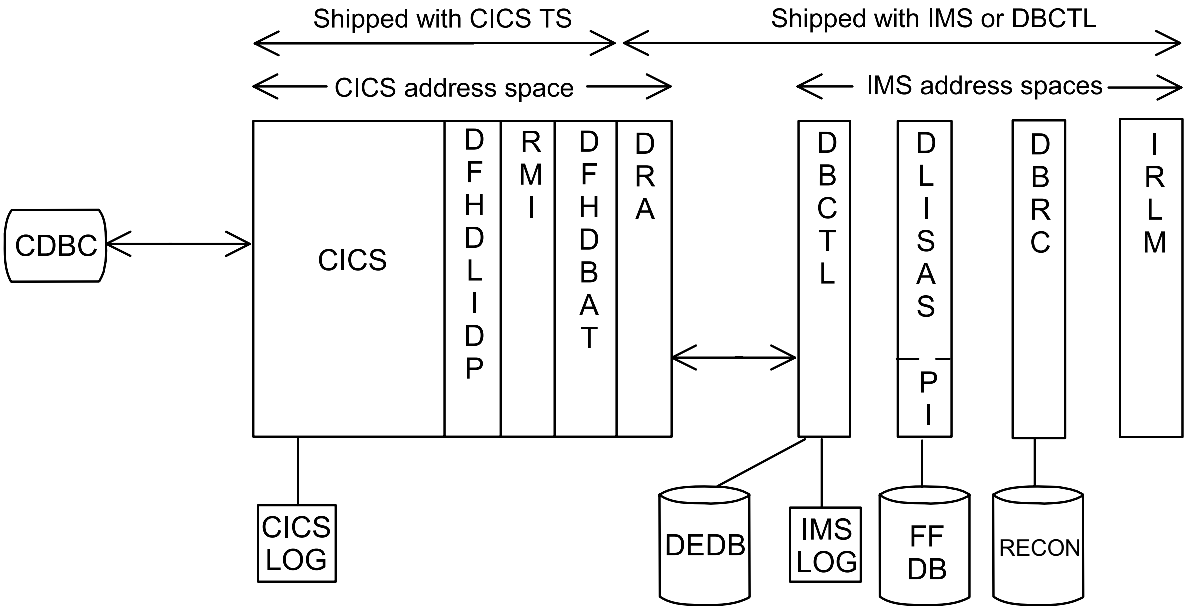


Figure 18-8. CICS-IMS DBCTL Environment CI207.0

Notes:

- For DBCTL access, actually there is an additional CICS interface module beyond the RMI, named DFHDBAT, which, however, is transparent both to system and application handling.
- The attachment component supplied by DBCTL is called the **Database Resource Adapter** (DRA).
- The DBCTL system and some characteristics of the connection between CICS TS and DBCTL are defined in the Database Resource Adapter (DRA) Startup Table.
  - CICS TS can connect to one DBCTL only at a time.
    Multiple DBCTL connections may be defined by different DRA Startup Tables.
- A separate DL/I address space (DLISAS) controls access to full-function databases.
- CICS applications may also share databases with IMS Batch Message Processing Programs (BMPs).
- All IMS logging, and all access to data entry databases, is under the control of IMS DBCTL.

System Setup Activities

Install/generate DBCTL,
define buffer pools, generate ACBs, and so on

On the CICS TS side:

- ✓ Assemble the DRA Startup Parameter Table
- ✓ Define and include CICS definitions
  - × Group DFHDBCTL in startup RDO list
  - × Destination CDBC for attachment error messages
  - × MCT entries from SDFHSAMP(DFH$MCTD)
  - × SIT parameters, particularly DBCTLCON=YES
  - × Add DBCTL (IMS) libraries to CICS JCL

Figure 18-9. System Setup Activities CI207.0

Notes:

- DBCTL has to be installed and verified, independently from CICS.
- Parts of the attachment are supplied by CICS TS, including user transactions and the PLT program for automatic attachment connection.
  - Resource definitions for these components are in RDO group **DFHDBCTL**.
  - DFHLIST contains an entry for DFHDBCTL.
- The attachment code writes its error messages to transient data queue **CDBC**, which is defined in CICS-supplied group **DFHDCTG**.
- The IMS **RESLIB** has to be added to the STEPLIB concatenation of the CICS startup JCL to be accessible by the attachment.

DRA Startup Parameter Table

DDNAME=	For dynamic allocation
DSNAME=	of the DRA RESLIB
DBCTLID=	IMSID from DBCTL startup
MINTHRD=	Number of threads to be activated automatically
MAXTHRD=	Maximum number of threads to be started if required
TIMER=	Time interval to retry connect if failed
TIMEOUT=	Number of seconds CICS waits for thread termination on CICS shutdown
...	Some buffer specifications

Figure 18-10. DRA Startup Parameter Table CI207.0

Notes:

- The DRA startup parameter table is built using IMS macro **DFSPRP** (supplied in `IMS.GENLIBB`), and is link-edited as module **DFSPZPxy**, normally into IMS RESLIB.
- The table's suffix has to be specified when starting the attachment (see next foil). The default suffix is '00'.
- Multiple DBCTL connections may be defined by different DRA startup tables.

Operating the DBCTL Attachment

- **CDBI** Inquire attachment state and DBCTL sysid if connected
- **CDBC** Start and stop attachment
- **CDBM** Enter commands to the DBCTL address space connected

- Automatic start of attachment through PLT or SIT specifications

```
DFHPLT TYPE=ENTRY,PROGRAM=DFHDELIM
DFHPLT TYPE=ENTRY,PROGRAM=DFHDBCON
```

SIT
```
INITPARM=(...,DFHDBCON='XY,IMSA',....)
DBCTLCON=YES
```


Figure 18-11. Operating the DBCTL Attachment CI207.0

Notes:

- CICS-supplied group DFHDBCTL defines operating transactions with a BMS user interface.
  - CDBI and CDBC control attachment functions within the CICS address space only.
  - The use of the DBCTL command transaction CDBM requires PSB DFHDBMP being active on the DBCTL side, introduced with DBCTL/IMS DM Version 5.
- PLT-invoked program DFHDBCON will use the DRA parameter table suffix specified in the SIT/INITPARM parameter.
  If this is not specified, it will attempt to load module DFSPZP**00**.
- `DBCTLCONN=YES` can be coded in the SIT instead of using the PLT; INITPARM specifications will take effect in the same way as for the PLT program.
- When starting the attachment with CDBC manually, INITPARM specifications may be overridden.
- There is no DBCTL-related program for shutdown PLT processing.
  On a regular shutdown of CICS TS, the DBCTL attachment always tries to quiesce its active threads for the time specified in the DRA startup parameter table.

CDBM Command File

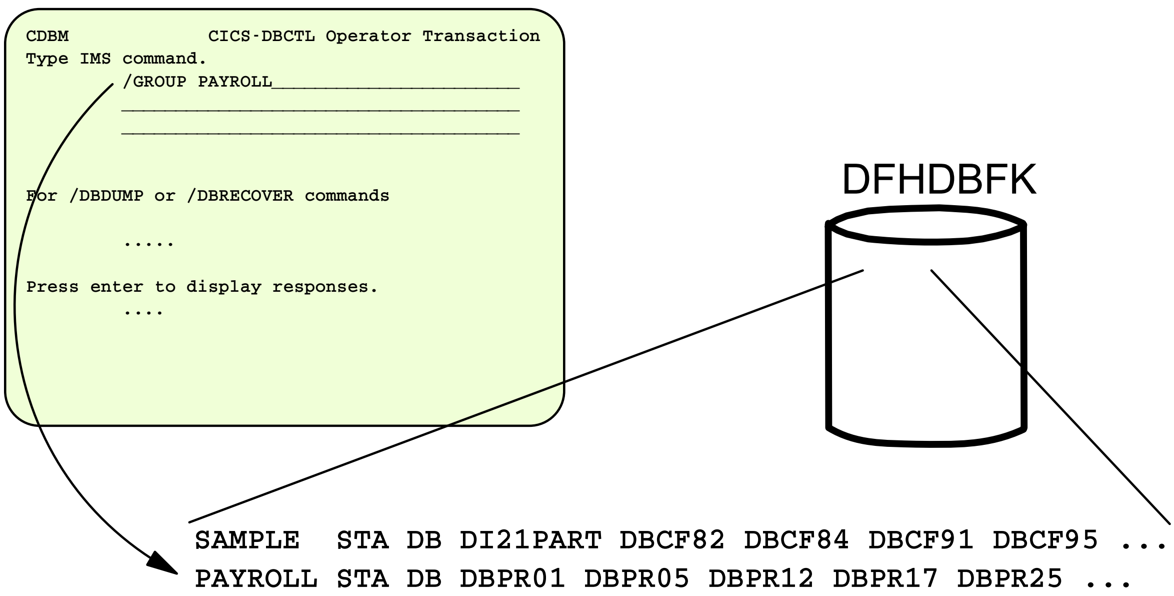


Figure 18-12. CDBM Command File CI207.0

Notes:

- The **CDBM** CICS-supplied transaction is the CICS-DBTCL operator transaction. It allows DBTCL (IMS) commands to be typed in their native form, and command responses to be received from DBCTL. The command input, however, is limited to four lines of 60 characters each.
- CICS TS 1.3 introduced a new CICS system file, DFHDBFK, which is used by the CDBM transaction to provide a repository for stored groups of DBCTL commands.
- Commands placed in the DFHDBFK file may contain an IMS command parameter field that is up to 1406 bytes in length.
- IMS commands are prefixed by a “group name”, which is the key field for each entry, and is used to select a specific entry on the `/GROUP` command keyed from the CDBM input screen.
- CBKF entries may be set up interactively using a special CDBM maintenance option (PF2), or offline by use of a batch job.

18.3 Accessing DB2 (SQL) Data

CICS DB2 Interface Overview

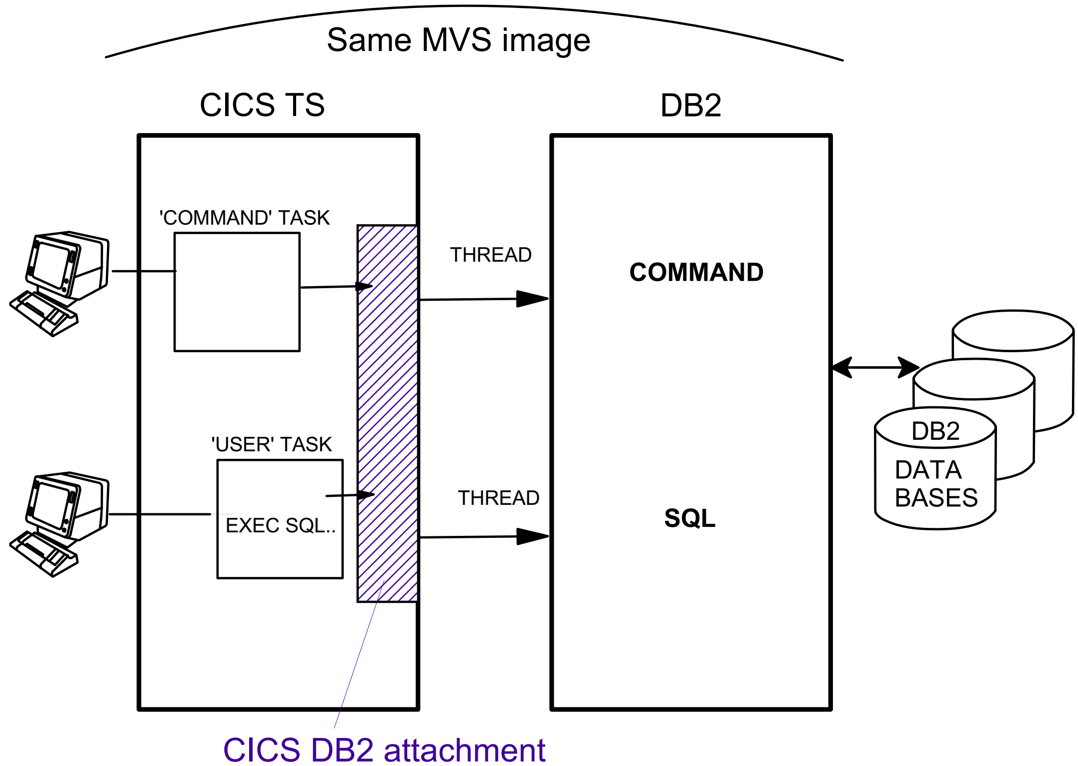


Figure 18-13. CICS DB2 Interface Overview CI207.0

Notes:

- CICS applications access DB2 data via EXEC SQL commands.
- Modules and macros that make up the CICS-DB2 attachment for CICS TS are integrated with the CICS Transaction Server for OS/390 product.
- A CICS TS region can be connected to one DB2 at a time. Definitions for connecting to different DB2 systems at different times may be prepared.
- Again, “communication” occurs via **threads**, each active thread being a separate MVS subtask.

  The CICS DB2 attachment delivered in CICS TS 2.2 has been enhanced, as it is now able to exploit the CICS open transaction environment (OTE) if the DB2 is at Version 6 or later. This reduces or even eliminates TCB switches on DB2 requests from CICS, which may considerably improve performance. This is especially interesting for enterprise beans running in CICS that make DB2 requests.
- You can enter DB2 commands from a CICS terminal, using dedicated *command threads* for processing.

Application Preparation

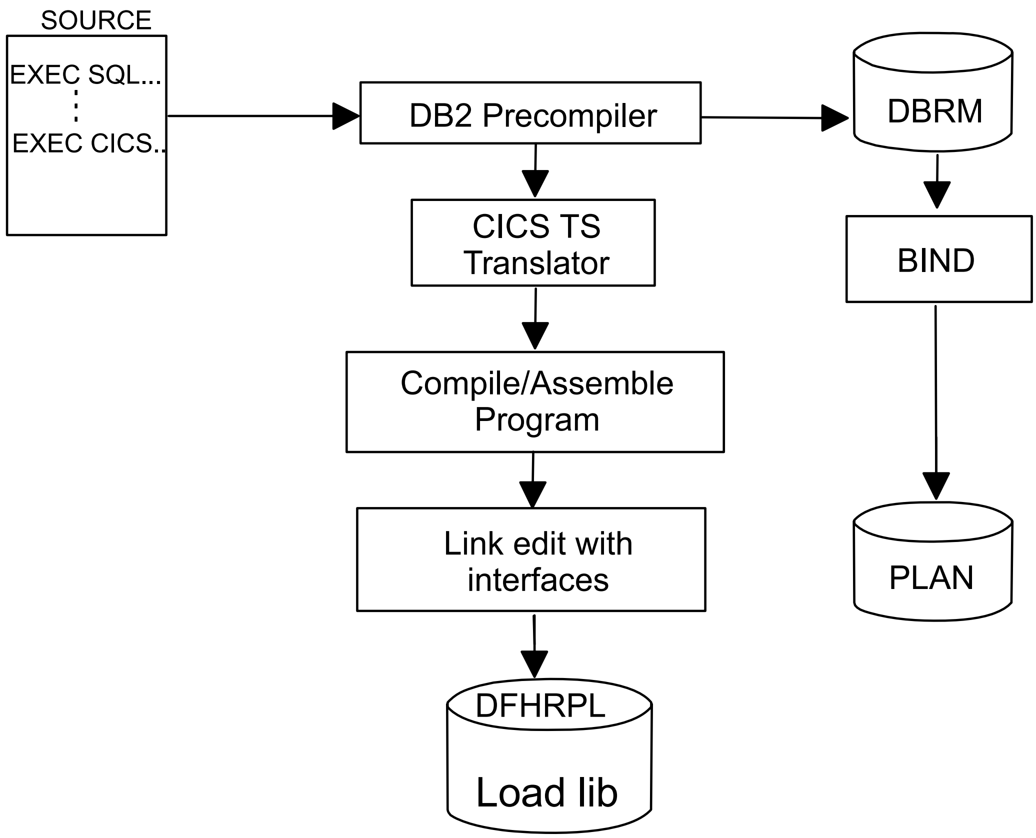


Figure 18-14. Application Preparation CI207.0

Notes:

The supplied procedures for compiling application programs may be modified to support DB2 by:

1. Adding a step for the DB2 precompiler. This step creates a Data Base Request Module (DBRM) containing information about each precompiled SQL statement.
2. Modifying the link step to include:

   DFHExI The CICS command level stub, where x denotes the language
   DSNCLI The DB2 interface

3. Adding a final step to **BIND** the DBRM into a **PLAN**.
   BINDing establishes the relationship between a program and its DB2 tables.
   A PLAN may consist of one or more DBRM members.

   Note that the latest enterprise versions of the IBM COBOL and PL/I compiler products provide integration of the CICS translator and the DB2 precompiler, respectively.

Types of Threads

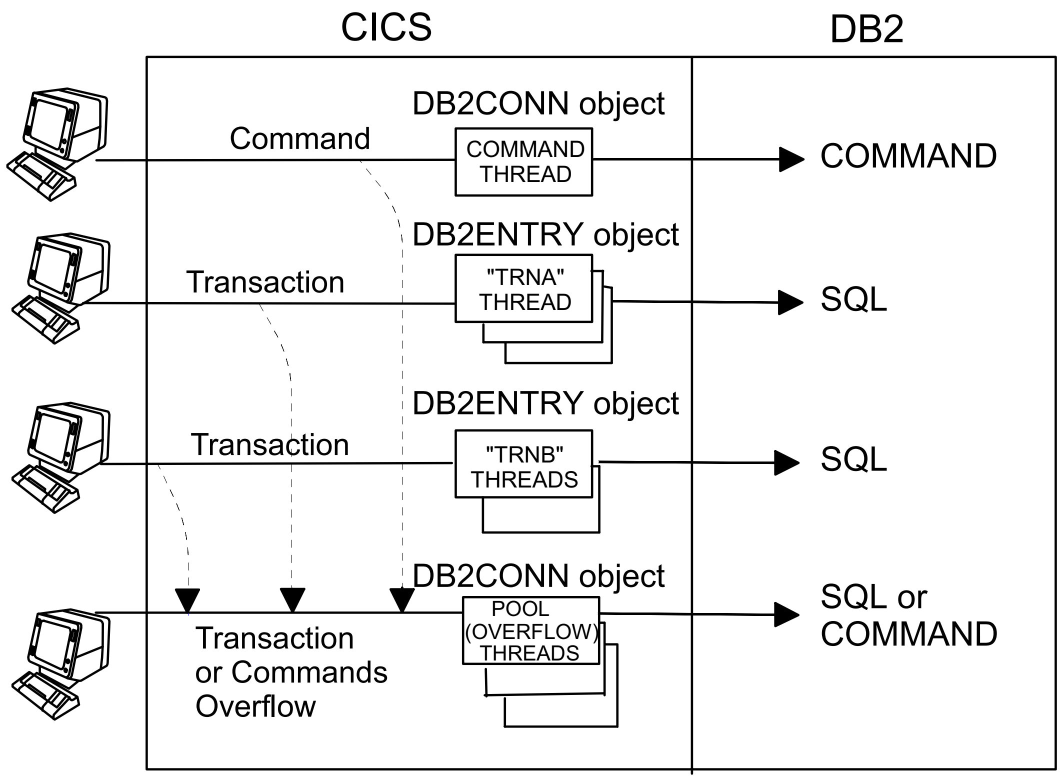


Figure 18-15. Types of Threads CI207.0

Notes:

Threads have to be defined through RDO by use of the resource types **DB2CONN** and **DB2ENTRY** resource definition object types.

There are three “types” of threads:

COMD *One* command thread is automatically defined and requires no explicit resource definition. It can *only* be used to support the execution of DB2 commands.

ENTRY Entry threads are referred to as **dedicated** threads. You define the number of threads and which specific CICS transaction ID(s) will use those threads. In this visual there are two groups of dedicated threads.

POOL For all transactions not having dedicated threads, and overflow from either command or dedicated threads.

CICS DB2 RDO Definitions

```
DEFINE DB2CONN(DB2P) GROUP(GRDB2P)
          DB2id        ==> DB2P
          DB2Groupid   ==> ....
          SIgnid       ==> CICSP
          PLAN         ==> POOLPLAN
          THREADLimit==> 0003
          AUTHType     ==> Sign
DEFINE DB2ENTRY(TRNTRANS) GROUP(GRDB2P)
          TRansid      ==> TRNA
          AUTHType     ==> Sign
          PLAN         ==> DSN8COPA
          PLANExitname  ==>
          THREADLimit==> 0003
          THREADWait   ==> Pool
DEFINE DB2TRANS(TRNB) GROUP(GRDB2P)
          Entry        ==> TRNTRANS
          TRansid      ==> TRNX
```


Figure 18-16. CICS DB2 RDO Definitions CI207.0

Notes:

Resource definitions for DB2 Attach are implemented as three RDO resource types:

DB2CONN Is a replacement for the RCT `TYPE=INIT`, `TYPE=COMD`, and `TYPE=POOL` definitions, defining the global attributes of the connection as well as the attributes of the pool threads and command threads to be used.

In CICS TS Version 2, support for the DB2 group attach has been introduced. Instead of naming a specific DB2 subsystem (`DB2id`), a DB2 Data Sharing Group (`DB2Groupid`) may be specified, and the connection will be to any member of the group that is active on the local MVS image.

DB2ENTRY Defines resources to be used by a group of one or more CICS transactions.

CICS transaction codes have to be correlated to DB2 plan names by `TYPE=ENTRY` statements, either by a static specification of a plan name, or by naming a user exit program that is to determine the plan name dynamically at execution time.

- On the first DB2 request of a UoW, the program selects the PLAN used by DB2.
- For dedicated threads, the allocated PLAN cannot be changed across a SYNCPOINT.
- With POOL threads, the PLAN may be changed across a SYNCPOINT.

One transaction is specified in the `TRANSID` option of the DB2ENTRY definition.

DB2TRAN Specifies an additional transaction to use the resources as defined in the named DB2ENTRY definition.

Thus, DB2ENTRY and DB2TRAN definitions together replace the RCT `TYPE=ENTRY`.

- The names used for the RDO objects are arbitrary strings, with no functional effects.
- Transaction names specified in DB2ENTRY and DB2TRANS may be generic, containing “**+**” or “*” wildcard characters.

Attachment Resource Definitions

- **Task exit interface definitions:**
  (only needed if DFHLIST is not used)

  ADD G(DFHRMI) LIST(mylist)

- **DB2 attachment resource definitions:**
  (only needed if DFHLIST is not used)

  ADD G(DFHDB2) LIST(mylist)

```
DEFINE TRANSACT(DSNC) PROG(DFHD2CM1)...
DEFINE PROGRAM(DFHD2CM1) LANG(Assem)
DEFINE PROGRAM(DFHD2CM2) LANG(Assem)
....
```

- **optional definitions:**

  ADD G(DFH$DB2) LIST(mylist)

```
DEFINE TRANSACT(DISP) PROG(DFHD2CM1)...
DEFINE TRANSACT(-DIS) PROG(DFHD2CM1)...
DEFINE TRANSACT(-STA) PROG(DFHD2CM1)...
```


Figure 18-17. Attachment Resource Definitions CI207.0

Notes:

- All resource definitions for the attachment objects are supplied in group **DFHDB2**, which is supplied in the `DFHLIST` structure.
- **DSNC** is the user transaction that comes with the attachment, which may be used to issue commands both for attachment control and against a connected DB2 system.
  - All these commands may be issued using DSNC followed by the appropriate command syntax.
  - In addition to the resource definitions defining the application resources of the CICS-DB2 sample application, IBM-supplied group **DFH$DB2** contains transaction definitions specifying the first four characters of attachment or DB2 commands as the transaction code. The program referred to is always DFHD2CM1.
  - DB2 commands entered from a CICS terminal are always prefixed with the “-” character, regardless of the actual command recognition character (CRC) or command prefix string (CPF) that is defined for this DB2.

CICS TS Improvements to the DB2 Attachment

- Open transaction environment
  - Application may block its TCB, without an impact to others, as TCB is for the sole use of a particular task.
  - Used by DB2 attachment, reduces TCB switches
- Threadsafe programs
  - Applications that use only threadsafe commands
  - May run completely under an open (L8) TCB, with potentially high positive effect on performance
- DB2 Group Attachment feature
  - DB2 Data Sharing group name usable for connection from CICS
- Enhanced DB2 Restart-Light supported
  - Cross-system restart in the event of an MVS system failure, in order to release retained locks and resynchronize indoubt units of work
- JDBC/SQLJ support for CICS applications
  - Standard DB2 Java API usable for all CICS Java applications:
    - Java Database Connectivity (JDBC) uses the dynamic SQL model
    - SQL Java (SQLJ) uses the static SQL model

Figure 18-18. CICS TS Improvements to the DB2 Attachment CI207.0

Notes:

Significant improvements to the CICS DB2 attachment have been provided by the latest releases of CICS TS in the areas of performance and of usability and standardization with DB2 capabilities.

For details, refer to the *CICS DB2 Guide, SC34-6252.*

Starting and Stopping the Attachment

Starting

- SIT — `INITPARM=(..,DFHD2INI='DSNP')` `DB2CONN=YES`
- PLTPI — `DFHPLT TYPE=ENTRY,` `PROGRAM=DFHD2CM0`
- Transaction (Command) — `DSNC STRT` `INITPARM` values; `DSNC STRT DSNP` overrides INITPARM

Stopping

- Automatically — at CICS shutdown
- Transaction — `DSNC STOP QUIESCE | FORCE`

Figure 18-19. Starting and Stopping the Attachment CI207.0

Notes:

- Before any DB2 activity can occur, the connection must be started:
  a. Automatically at initialization via program DFHD2CM0 specified in the Program List Table for startup after the DFHDELIM entry.
  b. SIT parameter `DB2CONN=YES|NO`, which has the same effect as the PLT entry when set to YES.
  c. Including the DSNC command as sequential terminal input. The connection will be created after the CONTROL IS GIVEN TO CICS startup message.
  d. By a CICS terminal operator entering the DSNC STRT command.
- When CICS shuts down, the attachments disconnects automatically.
- Be careful using the FORCE option on DSNC STOP; this will probably abend threads, and may subsequently require major recovery actions in order to maintain data integrity.
- The attachment supplied with CICS TS includes an *Alert Monitor* for automatic reconnect after DB2 goes down and restarts.

18.4 Accessing WebSphere MQ (MQSeries) Message Queues

The WebSphere MQ CICS Adapter (Overview)

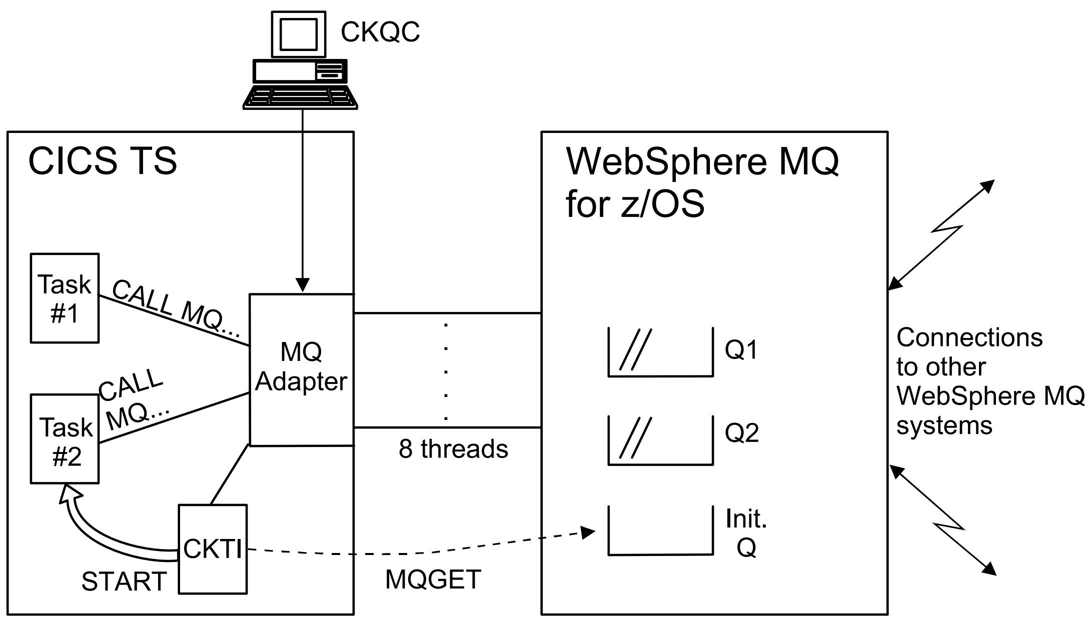


Figure 18-20. The WebSphere MQ CICS Adapter (Overview) CI207.0

Notes:

- The ERM attachment for CICS interfacing WebSphere MQ for z/OS is called an *adapter*. The CICS MQ adapter:
  - Is made up of transactions, programs, and map sets that are delivered with WMQ for z/OS
  - Allows CICS programs to access WMQ for z/OS objects using the Message Queuing Interface (MQI)
  - Provides a ready-to-use trigger mechanism that allows CICS transactions to be started when a trigger criterion is met by a MQ message queue
  - Contains a BMS-based command interface to display and control the connection between the CICS TS region and an instance of MQ
- A CICS TS region can be connected to only one MQ system (queue manager) at a time.
- CICS TS and the MQ Queue Manager must operate in the same MVS.
- Communication is by threads using MVS cross-memory services.
- The CICS MQ adapter supports a fix number of eight threads, allowing the appropriate number of MQ calls to be served at a time.

Preparing CICS WMQ Applications

Use regular CICS procedures (DFHYIT*) with additional specifications:

- SYSLIB concatenation of the compile step:
  - Add WebSphere MQ language support libraries for expanding MQ structures and constants:
    `SCSQASM, SCSQCOB, SCSQC370, SCSQPLI`

- Linkedit step:
  - Add the following JCL to access the MQ program stub:

```
//LKED.CSQSTUB DD DSN=MQS.SCSQLOAD,DISP=SHR
//LKED.SYSIN DD *
 INCLUDE CSQSTUB(CSQCSTUB)
 NAMEpgmname(R)
```


Figure 18-21. Preparing CICS WMQ Applications CI207.0

Notes:

- As the MQI is a call interface, no special precompile has to be performed.
- For use of the MQI, application programs will refer to MQ structures that have to be made available to the compile step.
- To interface with the attachment, CICS programs using the MQI have to be link-edited with the MQ-supplied program stub `CSQCSTUB`.

MQ Adapter CICS Resources

- WebSphere MQ supplies input to DFHCSDUP
- Three members in `MQS.SCSQPROC`:
  - `CSQ4B100` CICS MQ adapter resource definitions (required)
  - `CSQ4S100` Sample application resource definitions (optional)
  - `CSQ4D100` Distributed Queuing facility resource definitions (optional feature)
- CSQ4B100 specifies group CSQCAT1 containing ...
  - `CKQC` Adapter control interface transaction
  - `CKTI` Trigger Monitor transaction
  - `CKAM` Alert Monitor transaction
    and others

Figure 18-22. MQ Adapter CICS Resources CI207.0

Notes:

- Resource definitions for the MQ adapter have to be defined using the DFHCSDUP batch utility program.
- Input is supplied (ready-to-use) as members of the MQ for z/OS SCSQPROC library.
- The adapter definitions are required to support MQI applications; the other ones are optional.
- **CKQC** is the CICS transaction that invokes the MQ adapter display and control function.
- One or more instances of **CKTI** may be started as trigger monitors, each against a separate initiation queue.
- If MQ for z/OS is not available when the attachment is started from the CICS side, or the MQ queue manager goes down, the Alert Monitor transaction **CKAM** is automatically started, and will detect when MQ comes up (again) in order to (re-)connect without operator activity required.

WebSphere MQ - CICS Bridge

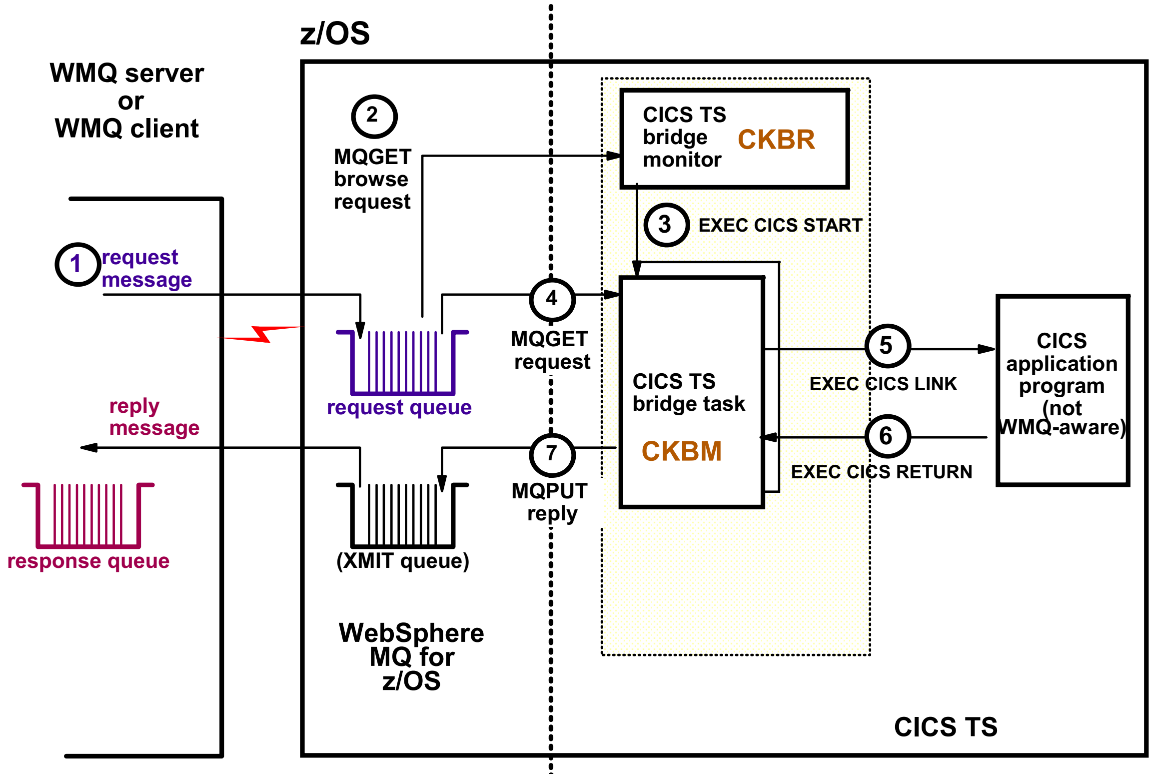


Figure 18-23. WebSphere MQ - CICS Bridge CI207.0

Notes:

The WQM-CICS bridge is an architected solution that enables a client WMQ application, which may run on any WMQ client or server platform, to interface with an existing CICS server application, without the need to apply any changes to that CICS application, and particularly without the need to add MQI calls to it.

- This CICS application may be a server program which is invoked via EXEC CICS LINK, or a transaction which is normally invoked from 3270 terminals.
- The WMQ for z/OS product supplies the CICS programs that act as the “bridge monitor” and “bridge task”, transforming the client’s request message contents to a CICS program or transaction invocation, and, in the other direction, creating an MQ reply message from the CICS application’s output.
- The client application has to provide its request within an architected message structure that contains fields to specify both the control input to the bridge components and the application data input to the target application. This is all the programming effort that has to be taken in order to integrate the CICS application.

Starting and Stopping the Adapter

Starting

- SIT based auto-connect `MQCONN=YES`
- SIT defined system parameters `INITPARM=(..,CSQCPARM='SN=CSQA,INITQ=CICS01.INITQ,TN=333',...)`
- PLTPI based auto-connect `DFHPLT TYPE=ENTRY, PROGRAM=CSQCCODF`
- CKQC transaction command interface (START option)

Stopping

- Automatically at CICS shutdown
- Transaction `CKQC` Stop option QUIESCE|FORCE

Figure 18-24. Starting and Stopping the Adapter CI207.0

Notes:

Before any MQI activity can occur, the connection must be started in one of two ways:

1. *Either*, automatically at initialization via SIT parameter `MQCONN=YES` or Program List Table entry for program CSQCCODF.
2. *Or*, by use of the CSQC adapter control transaction.

   Both methods refer to options specified within the SIT `INITPARM` parameter to identify the MQ subsystem and other subordinate specifications.

   The connection will be created, and the eight threads activated, after the `CONTROL IS GIVEN TO CICS` startup message.

- As there is no control table (as for DBCTL and DB2), the MQ system ID has to be determined either by INITPARM definition or via the CKQC input panel.
- On non-immediate shutdowns of CICS TS, the MQ adapter automatically quiesces active threads.
- Be careful when using the FORCE option to stop the adapter or perform an IMMEDIATE shutdown of CICS TS; this will probably abend active threads.

Accessing External RMs Data (Summary)

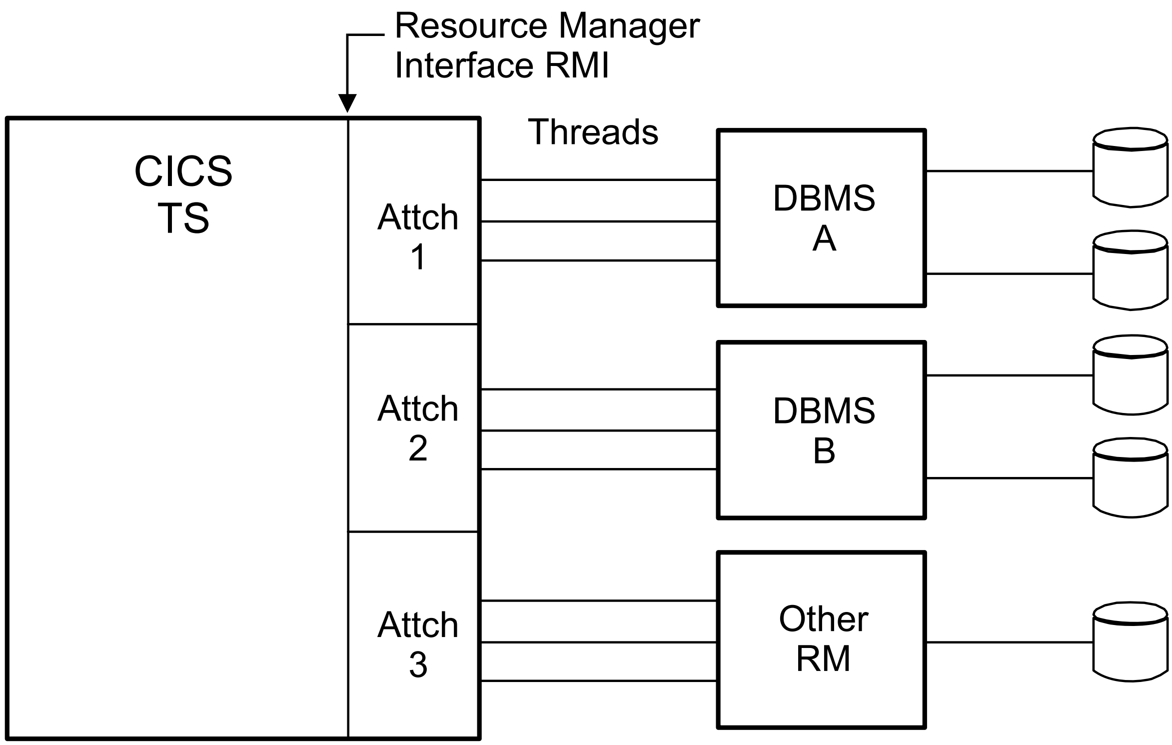


Figure 18-25. Accessing External RMs Data (Summary) CI207.0

Notes:

- CICS applications may access data that is owned and controlled by an External Resource Manager (ERM) via the CICS Resource Manager Interface.
- The connection and interaction between CICS TS and the ERM is managed by an attachment code that operates in the CICS address space as a Task Related User Exit.
- MVS cross-memory services are used for communication between CICS and the ERM, so CICS can only be directly attached to an ERM running in the same MVS image.
- Attachment code is supplied for the connection to DBCTL, DB2, and WMQ for z/OS.
- ERM attachments may provide a CICS user interface for attachment control.
- Generally, ERM attachments are started by a (Startup) PLT program in order to have the attachment active as soon as applications can be serviced within the CICS region.
- Application programs use the RM's APIs to access the external data. Therefore, an RM-specific program stub has to be linked with the CICS application code.
- Each Resource Manager performs logging and recovery actions only for its own resources.
- CICS TS coordinates unit-of-work processing with the ERMs by use of a two-phase commit protocol.

Unit Summary

Having completed this unit, you should be able to:

- Describe in general the terms RMI, TRUE, and RM stub
- State what IBM resource manager systems are providing support for the CICS RMI
- Describe the system and application setup activities that are required to access
  - DL/I data via Remote DL/I and DBCTL
  - SQL data via a DB2
  - WebSphere MQ message queues
- List the restrictions that exist for the named system interconnection options

Figure 18-26. Unit Summary CI207.0

Notes:

Appendix A. List of Abbreviations

ACB	Access Control Block
ACID	Atomicity; Consistency; Isolation; Durability
AKP	Activity Key Point
AOR	Application Owning Region
ATI	Automatic Transaction Initiation
BAS	Business Application Services
BDAM	Basic Direct Access Method
BMS	Basic Mapping Support
BRXA	Bridge Exit Area
BTS	Business Transaction Services
CEOT	CICS Operation Terminal Transaction
CFDT	Coupling Facility Data Table
CICS	Customer Information Control System
CMT	CICS Maintained Table
CORBA	Common Object Request Broker Architecture
CPSM	CICSPlex System Manager
CRLP	Card Reader / Line Printer
CTG	CICS Transaction Gateway
CWBA	CICS Web Bridge Analyzer
CWI	CICS Web Interface
CWS	CICS Web Support (supersedes CWI)
DASD	Direct Access Storage Device
DBRM	Database Request Module
DCT	Destination Control Table
DDM	Distributed Data Manager
DFHCSD	CICS System Definition File
DFHRPL	CICS Relocatable Program Libraries
DPL	Distributed Program Link
DRA	Database Resource Adapter
DSA	Dynamic Storage Area

ECI	External Call Interface
EIP	EXEC Interface Program
EMP	Event Monitoring Points
EPI	External Presentation Interface
ERM	External Resource Manager
ESM	External Security Manager
EXCI	External CICS Interface
FCP	File Control Program
FCT	File Control Table
FEPI	Front End Programming Interface
GEM	Global Enterprise Manager
GLUE(s)	Global User Exit(s)
GUI	Graphical User Interface
IIOP	Internet Inter-ORB Protocol
ISC	Inter System Communication
ISPF	Interactive System Productivity Facility
JVM	Java Virtual Machine
KSDS	Keyed Sequential Data Sets
LE	Language Environment
MCT	Monitor Control Table
MQI	Message Queuing Interface
MRO	Multi-Region Operation
ONC	Open Network Computing (ONC RPC)
OO	Object-Oriented
ORB	Object Request Broker
PCT	Processing Control Table (now TXD)
PD	Program Directory (was PPT)
PLT	Program List Table
PLTPI	Post Initialization Program List Table
PPT	Processing Program Table (now PD)
RCT	Resource Control Table
RDO	Resource Definition Online

RLS	Record Level Sharing
RMI	Resource Manager Interface
RPC	Remote Procedure Call
RRDS	Relative Record Data Sets
RRMS	Recoverable Resource Management Service
SIP	System Initialization Program (DFHSIP)
SIT	System Initialization Table
SNEX	Signon Extension (control block)
SOS	Short-on-Storage
TCA	Task Control Area
TCB	Task Control Block
TCT	Terminal Control Table
TCTTE	TCT Terminal Entries
TCTSE	TCT System Entries
TCTUA	TCT User Area
TDP	Transient Data Program
TIOA	Terminal I/O Area
TOR	Terminal Owning Region
TRUE(s)	Task Related User Exit(s)
TST	Temporary Storage Table
TTI	Terminal Transaction Initiation
TXD	Transaction Directory (was PCT)
UMT	User Maintained Table
UOW	Unit-of-Work
UR	Unit of Recovery
URM	User Replaceable Module(s)
WLM	Workload Manager
XLT	Transaction List Table
XM	Transaction Manager

Bibliography

Booklets:

- None

Manuals:

- *CICS Transaction Server for z/OS Version 3.1 Release Guide, GC34-6421*
- *Migration from CICS Transaction Server for OS/390 V1.3, GC34-6423*
- *Migration from CICS Transaction Server for z/OS V2.2, GC34-6424*
- *Migration from CICS Transaction Server for z/OS V2.3, GC34-6425*
- *CICS Transaction Server for z/OS Version 3.1 Installation Guide, GC34-6426*
- *CICS Transaction Server for z/OS Program Directory, GI10-2560*

The above titles are the only books provided automatically in hardcopy with CICS Transaction Server for z/OS, Version 2 Release 3. Several other books are available to order in hardcopy. Further information about the forms in which the published information for CICS is delivered may be found in the *CICS Transaction Server for z/OS Release Guide*, or the *CICS Transaction Server for z/OS Installation Guide*.

Administration

- *CICS System Definition Guide, SC34-6428*
- *CICS Customization Guide, SC34-6429*
- *CICS Resource Definition Guide, SC34-6430*
- *CICS Operations and Utilities Guide, SC34-6431*
- *CICS Supplied Transactions, SC34-6432*

Programming

- *CICS Application Programming Guide, SC34-6433*
- *CICS Application Programming Reference, SC34-6434*
- *CICS System Programming Reference, SC34-6435*
- *CICS Front End Programming Interface User's Guide, SC34-6436*
- *CICS C++ OO Class Libraries, SC34-6437*
- *CICS Distributed Transaction Programming Guide, SC34-6438*
- *CICS Business Transaction Services, SC34-6439*
- *Java™ Applications in CICS, SC34-6440*

Diagnosis

- *CICS Problem Determination Guide, SC34-6441*
- *CICS Messages and Codes, GC34-6442*
- *CICS Diagnosis Reference, LY33-6110*
- *CICS Data Areas, LY33-6107*
- *CICS Trace Entries, SC34-6443*
- *CICS Supplementary Data Areas, LY33-6108*

Communication

- *CICS Intercommunication Guide, SC34-6243*
- *CICS Family: Interproduct Communication, SC34-6267*
- *CICS Family: Communicating from CICS on System/390, SC34-6268*
- *CICS External Interfaces Guide, SC34-6244*
- *CICS Internet Guide, SC34-6245*
- *CICS Web Services Guide, SC34-6458*

Special topics

- *CICS Recovery and Restart Guide, SC34-6246*
- *CICS Performance Guide, SC34-6452*
- *CICS IMS Databate Control Guide SC34-6453*
- *CICS RACF Security Guide, SC34-6454*
- *CICS Shared Data Tables Guide, SC34-6250*
- *CICS DB2 Guide, SC34-6252*

Web URLs:

`http://www.ibm.net`

IBM's Internet Connection Web site

`http://www.ibm.com`

IBM Company Home Page

`http://www.ibm.com/cics`

IBM CICS Product Home Page

`http://www-306.ibm.com/software/htp/cics/txppacs/txpsumc.html`

IBM CICS Support Packs

Glossary

A

abbreviated trace. Optional format for CICS trace entries which summarizes the information in full trace entries. Each trace entry is described by a single line of text that is usually sufficient for debugging. Compare with full trace. See the CICS Transaction Server Problem Determination Guide for further information about trace.

abend. Abnormal end of task. An application can issue an EXEC CICS ABEND command to terminate a task abnormally.

abend code. See transaction abend code.

absolute time. In CICS interval control commands (START and POST), a time relative to the midnight before the time of issuing of the command (the current time). An absolute time earlier than the current time by less than six hours is interpreted as the current time. An absolute time earlier than the current time by more than six hours is interpreted as being relative to the midnight following the current time. For example, if the current time is 15:00, an absolute time of 11:00 is interpreted as 15:00, and an absolute time of 07:00 is interpreted as 07:00 tomorrow. As returned by an EXEC CICS ASKTIME command and input to an EXEC CICS FORMATTIME command, the number of milliseconds since 00.00 on 1 January 1900.

ACB. See access method control block (ACB).

access. The ability to read or update a resource. In CICS Transaction Server, access to protected resources is controlled by RACF® or an equivalent external security manager (ESM).

access authority. An authority that relates to a request for a type of access to protected resources. In RACF, the access authorities are: NONE, EXECUTE, READ, UPDATE, CONTROL, and ALTER.

access control environment element (ACEE). In RACF, a control block containing details of the current user, including user ID, current connect group, user attributes, and group authorities. An ACEE is constructed during user identification and verification.

accessibility. Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use software products successfully.

access intent. (1) In RACF, a subsystem's intended use of a protected resource. (2) In IMS, a subsystem's intended use of a database. This is in contrast to the sharing level of the database itself, which specifies how the database can be shared.

access key. In ESA key-controlled storage, a key associated with a storage access request. When key-controlled protection applies to a storage access, a store operation (write) is permitted only when the storage key matches the access key; a fetch (read) is permitted when the keys match or when the fetch-protection bit of the storage key is zero. In most cases, the access key for a storage operation is the program status word (PSW) key in the current PSW. For information about how ESA determines when access keys match storage keys, see the IBM Enterprise Systems Architecture/390 Principles of Operation manual.

access list. In RACF, a list within a profile of all authorized users and their access authorities. Synonym for standard access list. See also conditional access list.

access method. A technique for moving data between main storage and input/output devices, for example, VSAM, VTAM.

access method control block (ACB). A control block that links an application program (for example, a CICS system) to an access method (for example VSAM or VTAM). In communication with DL/I, an ACB is used only when the underlying access method is VSAM.

access method services (AMS). A utility program for the definition and management of VSAM data sets.

ACEE. See access control environment element (ACEE).

ACID properties. The term, coined by Haerder and Reuter [1983], and used by Jim Gray and Andreas Reuter to denote the properties of a transaction: Note: In CICS, the ACID properties apply to a unit of work (UOW). See also unit of work (UOW).

active task. A CICS task that is eligible for dispatching by CICS. During emergency restart, a task that completed an LUW and started another, but that did not cause any records to be written to the system log during the second LUW. During recovery-control processing, an LUW completion but no physical end-of-task (that is, task DETACH) is found.

activity keypoint. In CICS, a record of task status on the system log made on a periodic basis to facilitate the identification of transaction backout information during emergency restart. In the event of an uncontrolled shutdown and subsequent emergency restart, activity keypoints can shorten the process of backward scanning through the system log. Activity keypoints are written automatically by the system, as system activity keypoints, or by the user, as user activity keypoints.

AD/Cycle® Language Environment/370 (AD/Cycle). An SAA runtime library that establishes a common execution environment for a number of SAA programming languages. See also Systems Application Architecture (SAA).

address space. A range of up to two gigabytes of contiguous virtual storage addresses that the system creates for the user. Unlike a data space, an address space contains user data and programs, as well as system data and programs, some of which are common to all address spaces. Instructions execute in an address space, not a data space. Contrast with data space.

addressing mode (AMODE). The mode, 24-bit or 31-bit, in which a program holds and processes addresses. The AMODE linkage-editor control statement specifies the addressing mode of the load module produced. A program attribute that refers to the address length that a program is prepared to handle on entry. Addresses may be either 24 bits or 31 bits in length. In 24-bit addressing mode, the processor treats all virtual addresses as 24-bit values: in 31-bit addressing mode, the processor treats all virtual addresses as 31-bit values. Programs with an addressing mode of ANY can receive control in either 24-bit or 31-bit addressing mode. A control statement that defines the addressing mode of the load module produced by the linkage editor.

Advanced Program-to-Program Communication (APPC). The SNA protocol boundary of the presentation services layer of the LU6.2 architecture.

AFCB. See authorized function control block (AFCB).

after image. A record of the contents of a data element after it has been changed. After images are used for forward recovery.

agent. In a two-phase commit syncpointing sequence (LU6.2 or MRO), a task that receives syncpoint requests from an initiator (a task that initiates syncpointing activity).

AID. See automatic initiate descriptor (AID).

AIX®. See alternate index (AIX).

allocation. The assigning of various types of programs and record categories to system storage locations, such as main storage or disk storage.

alternate index (AIX). For VSAM key-sequenced data sets and entry-sequenced data sets, an index of alternate keys that provides a path for secondary access to the data set. If the records have alternate keys, the alternate index is built when the data set is created. A subordinate index in a hierarchy of indexes.

alternate index base data set (AIX VSAM). The VSAM data set that is the base or normal route of file access in a VSAM alternate index arrangement.

alternate key. In VSAM, a field, other than the primary key, of fixed length and position in a record. A set of alternate keys is used to build an alternate index that provides an alternative or secondary path for access to the data set. There can be any number of alternate keys in a record and they need not be unique.

alternate screen size. An option that permits the size of a display screen to be defined differently from the standard size.

American National Standard Code for Information Interchange (ASCII). An interchange code in which the code pages consist of 7-bit coded characters (8 bits including parity check). IBM has defined an extension to the ASCII code, using 8-bit coded characters.

AMODE. See addressing mode (AMODE).

AOR. See application-owning region (AOR).

APAR. See authorized program analysis report (APAR).

APF. See authorized program facility (APF).

application identifier (VTAM applid). The name by which a logical unit is known in a VTAM network.

The CICS applid is specified in the APPLID system initialization parameter.

application program. A program written for or by a user that applies to the user's work. In this sense, an application program is part or all of the implementation of an application. In data communication, a program used to connect and communicate with stations in a network, enabling users to perform application-oriented activities.

application programming interface (API). In CICS, the command-level programming interface supported by CICS for user application programs. For programming information, see the Application Programming Reference manual and the System Programming Reference manual. The formally-defined programming language interface between an IBM system control program or a licensed program and the user of the program.

application unit of work. A set of actions within an application that the designer chooses to regard as an entity in its own right. The designer decides how (if at all) an application should be subdivided into application units of work, and whether any application unit of work shall consist of just one or of many CICS logical units of work (LUWs). Typically, but not exclusively, an application unit of work would correspond to a CICS transaction.

application-owning region (AOR). In multiregion operation (MRO) or intersystem communication (ISC), a CICS address space whose primary purpose is to manage application programs. It receives transaction routed requests from a terminal-owning region (TOR). In a configuration that does not have a data-owning region (DOR), the AOR may contain file-related resources. See also data-owning region (DOR) and terminal-owning region (TOR).

asynchronous. An event that occurs at a time that is unrelated to the time at which another event occurs. The two events are mutually asynchronous. The relationship between the times at which they occur is unpredictable.

ATI. See automatic transaction initiation (ATI).

attach. In programming, to create a task that can execute concurrently with the attaching code.

attention identifier (AID). Part of the data stream sent to the host by a 3270 terminal, indicating which PF key or PA key (including ENTER, CLEAR, and so on) caused the data to be sent to the host.

authorized program analysis report (APAR). A report on the basis of which IBM supplies a fix of a temporary corrective nature to elements of function SYSMODs. APAR fixes are called "corrective" service because they are installed to cure problems currently being experienced by an installation. The APAR fix is usually in the form of either a modification to a load module or an update to card-image data. It is intended as a temporary arrangement until a preventive service (PTF) is issued to fix the problem permanently. This PTF supersedes the APAR fix, and indeed specifies this relationship on its ++VER statement.

authorized program facility (APF). A facility that enables identification of programs that are authorized to use restricted functions.

autoinstall. A method of creating and installing resources dynamically as terminals log on, and deleting them at logoff. Autoinstall can be used for VTAM terminals, MVS consoles, APPC connections, programs, map sets, partitionsets and journals.

autoinstall control program. A user-replaceable CICS program used to select some of the data needed to automatically install terminals, notably the CICS terminal identifier (TERMID) and the model name to be used in each instance. For programming information, see the Customization Guide.

automatic initiate descriptor (AID). A control block used internally by CICS for scheduling purposes. An example of AID use is scheduling a transaction, optionally associating it with a terminal and a temporary storage queue. Another use is scheduling MRO, LU6.1, and LU6.2 ALLOCATE requests.

automatic transaction initiation (ATI). The initiation of a CICS transaction by an internally-generated request, for example, the issue of an EXEC CICS START command or the reaching of a transient data trigger level.

CICS resource definition can associate a trigger level and a transaction with a transient data destination. When the number of records written to the destination reaches the trigger level, the specified transaction is automatically initiated. See the appropriate Resource Definition manual.

Any CICS application can issue an EXEC CICS START TRANID command to start a named transaction immediately, at a specified time, after a specified delay, or, if a terminal is required, as soon as that terminal is free. See the Application Programming Reference manual.

auxiliary storage. Data storage other than main storage; for example, storage on magnetic tape or direct access devices.

auxiliary trace. An optional CICS function that causes trace entries to be recorded in the auxiliary trace data set, a sequential data set on disk or tape.

B

back-end transaction. In synchronous transaction-to-transaction communication, a transaction that is started by a **front-end transaction**.

backout. The process of restoring to a previous state all or part of a system. The process of removing all the updates against protected resources such as files and DL/I databases performed by an application program that either has terminated abnormally or was inflight at the time of a CICS or MVS or VSE image failure. Backout can by done dynamically in the case of an application abend, or during restart in the case of CICS or MVS or VSE failure.

backup. The process of making a copy of a data file that can be used if the original file is destroyed.

backup copy. A copy, usually of a file or a library member, that is kept in case the original file or library member is unintentionally changed or destroyed.

backup-while-open (BWO). A facility that allows a backup copy of a VSAM data set to be made while the data set remains open for update.

When you take a backup-while-open (BWO) copy of a data set, only the updates that are made after the BWO need to be recovered in the event of a disk failure. This considerably reduces the amount of forward recovery that is needed.

backward recovery. The process of restoring integrity to databases and other recoverable resources following a failure. Before a change is made to an element of a recoverable resource, such as a file record or a database segment, a before image of the element is recorded. Before images are used to reverse the changes that were made by logical units of work that were incomplete when the failure occurred. The recoverable resource is thus returned to a known state and processing can continue.

base cluster. In systems with VSAM, a key-sequenced or entry-sequenced file over which one or more alternate indexes are built.

basic direct access method (BDAM). (CICS Transaction Server only.) An access method used to retrieve or update particular blocks of a data set on a direct access device.

basic mapping support (BMS). A facility that moves data streams to and from a terminal, providing device independence and format independence for application programs. BMS is an interface between CICS and its application programs. It formats input and output display data in response to BMS commands in programs. To do this, it uses device information from CICS system tables and formatting information from maps you have prepared for your application programs.

BMS provides message routing, terminal paging, and device independence services. Most of the BMS programs are resident in the CICS nucleus.

BMS exists in three pregenerated versions: minimum, standard, and full function Each version provides a different level of function, and therefore requires a different amount of virtual storage. The minimum version uses considerably less storage than the other two versions.

basic sequential access method (BSAM). An access method for storing or retrieving data blocks in a continuous sequence.

before image. A record of the contents of a data element before it is changed. Before images are used to backout incomplete or incorrect changes in the event of a failure.

BMS. See **basic mapping support (BMS)**.

BMS maps. Maps telling BMS how to format field data for display. They are not needed for text data output. Every BMS mapping command names a map that contains formatting (mapping) instructions. Each map has two forms: physical and symbolic.

BMS message routing. The routing of data to one or more terminals other than the originating terminal.

BTS. CICS business transaction services.

buffer. An area of processing storage used to hold a block of data while it is waiting to be processed or written to an I/O device.

business application. Any set of CICS resources that represent a meaningful entity to an enterprise or a user (such as, Payroll).

BWO. See **backup-while-open (BWO)**.

Business transaction. A self-contained business function, for example, the booking of an airline ticket.

Traditionally, in CICS a business transaction might be implemented as multiple user transactions; the booking of the airline ticket might be undertaken by transactions that inquire about availability, reserve the seat, deal with payment, and print the ticket, for example. Using BTS, a business transaction might be implemented as multiple activities.

C

C/370™. A programming language designed for a wide range of system and commercial applications.

CA. See control area (CA).

CA splitting. In VSAM, to double a control area dynamically and distribute its CIs evenly when the specified minimum of free space is used up by more data.

card reader/line printer (CRLP). A card reader and line printer. In CICS terminal control, a pair of input and output sequential data sets that simulate a card reader and line printer. See discussion of sequential (BSAM) devices in the CICS Transaction Server System Definition Guide or the CICS/VSE System Definition and Operations Guide.

CBIPO. See Custom-Built Installation Process Offering (CBIPO).

CBPDO. Custom-Built Product Delivery Offering (SMP/E).

CDT. See class descriptor table (CDT).

CEDA. A CICS transaction that defines resources online. Using CEDA, you can update both the CICS system definition data set (CSD) and the running CICS system. Compare with CEDB. See the CICS-Supplied Transactions manual for more information.

CEMT. A CICS transaction that invokes all the master terminal functions. These functions include inquiring and changing the value of parameters used by CICS, altering the status of system resources, terminating tasks, and shutting down CICS. See the CICS-Supplied Transactions manual for more information.

central electronic complex (CEC). One or more central processors running under the control of a single MVS or VSE operating system. The processor can be either a uniprocessor or a multiprocessor (including a dyadic processor).

CI. See control interval (CI).

CI splitting. In VSAM, to double control interval dynamically and distribute its records evenly when the specified minimum of free space is used up by new or lengthened records.

CICS. Customer Information Control System (CICS): IBM's general-purpose online transaction processing (OLTP) software is an e-business, industrial-strength, server for mission-critical applications. CICS is a powerful application server that runs on S/390 servers and a range of other operating systems, IBM and non-IBM, from the smallest desktop to the largest mainframe. It is used in Client/Server environments and in networks ranging in size from a few terminals to many thousands of terminals. It is a layer of middleware that seamlessly integrates all the basic software services required by OLTP applications together with a rich set of resources and management services in a highly available, reliable, and scalable manner, enabling its customers to concentrate on the tasks relevant to their particular business. Its application programming interface (API) enables programmers to port applications to and from a wide variety of hardware and software platforms where CICS is available, and because each product in the CICS family can interface with other members of the CICS family, this enables inter-product communication. Customers may write their own applications or choose from many existing vendor-written products. CICS is an IBM licensed program.

CICS dynamic storage area (CDSA). A storage area allocated from CICS-key storage below the 16MB line, intended primarily for the small amount of CICS code and control blocks that remain below the line in CICS Transaction Server 3.3. The size of the CDSA is controlled by the CDSASZE system initialization parameter.

CICS Internet Gateway. A workstation application that can accept requests from Web browsers and route them into CICS. It uses a CICS client and the EPI.

CICS messages and codes data set (DFHCMACD). A VSAM key-sequenced data set (KSDS) that is created and loaded by running the DFHCMACI job. Service changes can be applied to the DFHCMACD data set by running the DFHCMACU job. The CMAC transaction uses the DFHCMACD data set to provide online descriptions of CICS messages and codes.

CICS monitoring facility. The CICS component responsible for monitoring and producing task-related statistics information, such as task CPU usage and waits for I/O request units on an individual task basis. Reporting is divided into classes. See also accounting class data, performance class data, exception class data, and (for CICS Transaction Server only) SYSEVENT data.

CICS program library (DFHRPL). A library that contains all user-written programs and CICS programs to be loaded and executed as part of the online system. DFHRPL includes the control system itself and certain user-defined system control tables essential to CICS operation. The library contains program text and, where applicable, a relocation dictionary for a program. The contents of this library are loaded asynchronously into CICS dynamic storage for online execution.

CICS region userid. The userid assigned to a CICS region at CICS initialization. It is specified either in the RACF started procedures table when CICS is started as a started task, or on the USER parameter of the JOB statement when CICS is started as a job.

CICS segment. The portion of a RACF profile containing data for CICS.

CICS system. The entire collection of hardware and software required by CICS. See the Release Guide for information on hardware and software prerequisites.

In CICSPlex SM topology, a definition referring to a CICS system that is to be managed by CICSPlex SM.

CICS system definition data set (CSD). A VSAM KSDS cluster that contains a resource definition record for every record defined to CICS using resource definition online (RDO).

CICS web interface. A collection of CICS resources supporting direct access to CICS transaction processing services from Web browsers.

CICS-attachment facility. A facility that provides a multithreaded connection to DB2® to allow applications running under CICS to execute DB2 commands.

CICS-key. Storage protection key in which CICS is given control (key 8) when CICS storage protection is used. This key is for CICS code and control blocks. CICS-key storage can be accessed and modified by CICS. Application programs in user-key cannot modify CICS-key storage, but they can read it. CICS-key storage is obtained in MVS key-8 storage. Compare with user-key.

CICS-maintained data table (CMT). A type of CICS data table, for which CICS automatically maintains consistency between the table and its source data set. All changes to the data table are reflected in the source data set and all changes to the source data set are reflected in the data table.

CICS-value data area (CVDA). A CICS value on INQUIRE and SET commands, specifically those that refer to resource status or definition. See the System Programming Reference manual for more information.

CICSPlex®. A CICS complex. A collection of related and connected CICS regions, which helps to address the inefficiencies in having multiple, full-function CICS systems processing a single OLTP workload. Each region belonging to the CICSPlex performs a major subset of the CICS function and is known as a resource manager region. In a typical CICSPlex, the CICS function is distributed among Terminal Owning Regions (TORs), which manage terminal sessions; Application Owning Regions (AORs), which process transactions and route the results back to the originating TOR; and File Owning Regions (FORs), which manage VSAM and BDAM files and VSAM data tables. For more information, see CICSPlex SM Concepts and Planning, GC33-0786.

In CICSPlex SM, a management domain. The largest set of CICS regions, or systems, to be manipulated by CICSPlex SM as a single entity. CICS systems in a CICSPlex being managed by CICSPlex SM do not need to be connected to each other.

CICSPlex SM. CICSPlex System Manager/ESA (CICSPlex SM) is a system-management tool that enables you to manage multiple CICS systems as if they were one. CICSPlex SM can manage independent, full-function CICS systems running on one or more connected central processor complexes (CPCs) just as easily as it can manage multiple, interconnected CICS systems functioning as a CICSPlex, also on one or more connected CPCs. The key functions provided by CICSPlex SM include:

- Real-time display and update of operational data relating to multiple CICS systems and resources
- Collection of statistical data for specific CICS resources
- Automated workload management
- Automated exception reporting for CICS resources.

For more information, see *CICSPlex SM Concepts and Planning*, GC33-0786.

CICSVR. CICS VSAM Recovery MVS/ESA Version 2 Release 1 (program number 5695-010) provides forward recovery for VSAM data sets and batch backout of VSAM data sets used by CICS Transaction Server. It can accept backups taken using the backup while open (BWO) facility.

class descriptor table (CDT). In RACF, a table containing class descriptors. The CDT contains descriptors with default class names for CICS resources. Users can modify the supplied descriptors and add new ones.

COBOL. Common business-oriented language. An English-like programming language designed for business data processing applications.

code page. In computer character representation, a list of values (usually single-byte or double-byte values) and the characters that they represent. To cater for different languages and the character requirements of different types of application, an interchange code (such as EBCDIC) typically defines several code pages. See interchange code. A specification of code points for a graphic character set or in a collection of graphic character sets. Within a code page, each code point can have one specific meaning only.

cold start. The standard initialization sequence performed by the CICS system initialization program. In a cold start, all resource definitions are refreshed. Any resources dynamically installed by the CEDA transaction in a previous execution are lost. See the CICS Transaction Server System Definition Guide.

command (EXEC CICS). In CICS, an instruction similar in format to a high-level programming language statement. CICS commands usually begin with the keywords EXEC CICS, and can be issued by an application program to make use of CICS facilities. The CICS translator converts EXEC CICS commands into the source language before the program is compiled.

command security. A form of security checking that can be specified for the PERFORM, COLLECT, DISCARD, INQUIRE, and SET commands. Command security operates in addition to any transaction security or resource security specified for a transaction. For example if a terminal invokes a transaction that the user is authorized to use, and the transaction issues a command that the user is not authorized to use, the command fails with the NOTAUTH condition.

command-level interface. The high-level programming interface that uses CICS commands beginning with the verb EXEC. Synonymous with EXEC interface and CICS API.

COMMAREA. See communication area (COMMAREA).

common system area (CSA). A major CICS storage control block that contains areas and data required for the operation of CICS. See the Performance Guide for more information. In MVS, an area that contains system control programs and control blocks. The storage areas within the common area are the system queue area (SQA), the pageable link pack area (PLPA), the (optional) modified link pack area (MLPA), a pageable BLDL table, a copy of the prefixed storage area (PSA) (for multiprocessor systems only), and a common system area (CSA).

common user access (CUA®). In SAA, a standard specification for the design and use of screen elements and user interaction techniques. The use of CUA improves usability and facilitates the transfer of user skills between applications. (CICS Transaction Server only: See the CICS Transaction Server System Application Guide for a comprehensive description of how to design CUA-conforming programs that communicate with non programmable terminals using BMS.)

common work area (CWA). The common work area (CWA) is an area within the CSA that can be used by application programs for user data that needs to be accessed by any task in the system. This area is acquired during system initialization and its size is determined by the system initialization parameter, WRKAREA. The CWA is initially set to binary zeros, and its contents can be accessed and altered by any task during CICS operation. Contrast with transaction work area (TWA).

communication area (COMMAREA). A CICS area that is used to pass data between tasks that communicate with a given terminal. The area can also be used to pass data between programs within a task.

control area (CA). In VSAM, a group of control intervals used as a unit for formatting a data set before adding records to it. Also, in a key-sequenced data set, the set of control intervals, pointed to by a sequence-set index record, that is used by VSAM for distributing free space and for placing a sequence-set index record adjacent to its data.

control block. In CICS, a storage area used to hold dynamic data during the execution of control programs and application programs. Synonym for control area. Contrast with control table.

control interval (CI). (1) A fixed-length area of auxiliary-storage space in which VSAM stores records and distributes free space. (2) The unit of information transmitted to or from auxiliary storage by VSAM in a single operation, independent of physical record size.

conversation. In distributed transaction processing, a sequence of exchanges over a session, delimited by SNA brackets. A dialog between two programs in which each program alternately sends and receives data. A dialog between CICS and a terminal user in which CICS alternately accepts input and responds.

conversational. Pertaining to a program or a system that carries on a dialog with a terminal user, alternately receiving and transmitting data. Pertaining to an SNA conversation or a dialog between two programs.

cross-systems coupling facility (XCF). A facility of MVS/ESA SP 4.1 that provides some initial MVS services needed to support a multisystem environment while still maintaining a single system image. Systems coupled using XCF are known as a SYSPLEX.

CSA. See common system area (CSA).

CUA. See common user access (CUA).

CVDA. See CICS-value data area (CVDA).

CWA. See common work area (CWA).

D

daisy chain. In CICS intercommunication, the chain of sessions that results when a system requests a resource in a remote system, but the remote system discovers that the resource is in a third system and has itself to make a remote request.

DASD. See direct access storage device (DASD).

Data Language/I (DL/I). In CICS Transaction Server, a high-level interface between applications and IMS; in CICS/VSE, an IBM database management facility provided by the DL/I DOS/VS database licensed program. It is invoked from PL/I, COBOL, or assembler language by means of ordinary subroutine calls. DL/I enables you to define data structures, to relate structures to the application, and to load and reorganize these structures. It enables applications programs to retrieve, replace, delete, and add segments to databases.

data space. A range of up to two gigabytes of contiguous virtual storage addresses that a program can directly manipulate through ESA/370 instructions. Unlike an address space, a data space can hold only user data; it does not contain shared areas, system data, or programs. Instructions do not execute in a data space, although a program can reside in a data space as nonexecutable code. Contrast with address space.

data table. A file whose records are held in main storage. See also CICS-maintained data table and user-maintained data table.

data-owning region (DOR). A CICS address space whose primary purpose is to manage files and databases. See application-owning region (AOR), and terminal-owning region (TOR).

database. A collection of interrelated or independent data items stored together without redundancy to serve one or more applications. See also hierarchy.

Database Control (DBCTL). An interface between CICS Transaction Server and IMS/ESA that allows access to IMS DL/I full-function databases and to data entry databases (DEDBs) from one or more CICS systems without the need for data sharing. It also provides release independence, virtual storage constraint relief, operational flexibility, and failure isolation.

DATABASE 2 (DB2). A relational database management system in which data is presented to the user in the form of tables. It can be accessed by CICS application programs issuing SQL requests.

DD statement. Data definition statement in MVS JCL. A DD statement specifies the name and characteristics of a data set to be associated with a file definition in the FCT. The name of the DD statement is the same as the name of the file definition. A DD statement can specify a concatenation of several data sets.

ddname. In MVS JCL, the name of a DD statement. See DD statement.

deadlock. (1) Unresolved contention for the use of a resource. (2) An error condition in which processing cannot continue because each of two elements of the process is waiting for an action by, or a response from, the other.

default user. The user whose security attributes are used to protect CICS resources in the absence of other, more specific, user identification. For example, except in the case of terminals defined with preset security, the security attributes of the default user are assigned to terminal users who do not sign on.

destination. A queue of data used with the CICS transient data facility. See transient data.

DFH. Three-character prefix of all CICS modules.

DFHCSDUP. CICS system definition data set (CSD) utility program. It provides offline services for the CSD. In CICS Transaction Server only, it can be invoked as a batch program or from a user-written program running either in batch mode or under TSO.

direct access. A method for retrieval or storage of a VSAM data record that is independent of the record's location relative to the previously retrieved or stored data. Contrast with sequential access.

direct access storage device (DASD). A storage device that provides direct access to data.

dirty read. A read request that does not involve any locking mechanism, and which may obtain invalid data--that is, data that has been updated, but is not yet committed, by another task. This could also apply to data that is about to be updated, and which will be invalid by the time the reading task has completed.

For example, if one CICS task rewrites an updated record, another CICS task that issues a read before the updating task has taken a syncpoint will receive the uncommitted record. This data could subsequently be backed out, if the updating task fails, and the read-only task would not be aware that it had received invalid data.

dispatch (dispatching). To allocate time on a processor to jobs or tasks that are ready for execution. In CICS, to schedule a task for execution. Dispatching is done by CICS task control.

dispatcher domain. Major component of CICS concerned with attaching, running, and detaching tasks and scheduling task control blocks for the various modes: quasi reentrant, resource-owning, or concurrent.

dispatching priority. A number assigned to tasks, used to determine the order in which they are to use the processor in the CICS multitasking environment.

distributed program link (DPL). Function of CICS intersystem communication that enables CICS to ship LINK requests between CICS regions. See the Intercommunication Guide for a list of the CICS family products that support DPL.

distributed routing model. A "peer-to-peer" dynamic routing system, in which each of the participating CICS regions can be both a routing region and a target region. The distributed routing model is implemented by the distributed routing program.

distributed routing program. A CICS-supplied user-replaceable program that can be used to dynamically route:

BTS processes and activities

Transactions started by non-terminal-related EXEC CICS START commands.

distributed transaction processing (DTP). Type of intercommunication in CICS, in which the processing is distributed between transactions that communicate synchronously with one another over intersystem or interregion links.

DTP enables a CICS application program to initiate transaction processing in a system which supports LUTYPE6, and which resides in the same or a different processor system or in different regions of the same processor. For more information, see the Distributed Transaction Processing Guide.

distributed unit of work (DUW). In a distributed process, all processing between two syncpoints taken by two or more intercommunicating transactions using a two-phase commit protocol. A DUW is a distributed LUW.

DOCTEMPLATE. A CICS resource that defines the attributes of a document template.

document template. A unit of information that is used to construct a document. A document template can contain fixed text, and symbols that represent text whose value is supplied by an application program. Document templates can be created by a CICS application, or retrieved from an external source. For more information, see "Using the DOCUMENT programming interface".

domain gate. An entry point or interface to a CICS domain. A domain gate can be called by any authorized caller who needs to use some function provided by the domain.

DOR. See data-owning region (DOR).

double-byte character set (DBCS). A set of characters in which each character is represented by two bytes. Languages, such as Japanese, which contain more symbols than can be represented by 256 code points require DBCS.

DPL. See distributed program link (DPL).

DSA. See dynamic storage area (DSA).

DTB. See dynamic transaction backout (DTB).

DTP. See distributed transaction processing (DTP).

dump. A representation of the contents of selected areas of main storage used to find out whether a program is functioning as intended and to analyze problems.

Dumps may be recorded by CICS either as a consequence of failure detected during CICS execution, or upon explicit request. CICS allows you some flexibility in specifying the content and type of dump to be produced.

dump code. (1) In CICS Transaction Server, a predefined name by which a dump is known. There are two types of dump code, transaction dump codes and system dump codes, used in transaction dumps and system dumps, respectively. See transaction dump code and system dump code.

A dump code can be defined by CICS or the user and is used to select a set of system actions. These actions are held in either the system or transaction dump table (see transaction dump table and system dump table). The CICS Transaction Server Problem Determination Guide describes the set of default actions that are taken if an entry does not exist. The CICS Transaction Server Messages and Codes manual contains a description of the transaction and system dump codes defined by CICS.

dump data set. A sequential data set (optional) used to record dumps of transactions (tasks) within the system. It can be formatted and printed by the CICS dump utility program (DFHDUP). If required, the user can define two dump data sets (DFHDMPA and DFHDMPB), switching between them during online execution of CICS.

dump table. A table of dump codes to enable a user to vary the system actions taken when a dump is produced for a particular dump code. There are two dump tables-one containing system dump codes for system dump requests, and one containing transaction dump codes for transaction dump requests. See system dump code and transaction dump code.

Dump tables are internally maintained by CICS, but cannot be externally generated like CICS control tables. Table entries can be explicitly added, modified, or deleted during a CICS run by CEMT transactions or EXEC CICS commands. If an entry for a requested dump code does not exist, a set of default system actions are implicitly added to the table but are not written to the global catalog. New or changed explicitly added entries are written to the global catalog. During dump domain initialization on a warm or emergency start, the tables are loaded from the global catalog.

dump utility program (DFHDUP). An offline utility program that formats and prints the output from formatted dump, and prints transaction dumps. It operates in batch mode and, for formatted dumps, identifies each storage area, program, and table entry, and prints them separately, with actual and relative addresses.

dynamic allocation. Facility of IMS Version 2.2 (or later) and of CICS Transaction Server, for allocating DL/I databases and CICS file control data sets, respectively. If no DD statement is provided for the database data sets contained in the database, allocation happens automatically when the database is scheduled.

dynamic backout. A process that automatically cancels all activities performed by an application program that terminates abnormally. See also backout and syncpoint.

dynamic routing. The automatic routing of a transaction or program, at the time it is initiated, from a requesting region to a suitable target region. Routing terminal data to an alternative transaction at the time the transaction is invoked. To do this, CICS allows the dynamic routing program to intercept the terminal data and redirect it to any system and transaction it chooses.

dynamic routing program. A user-replaceable CICS program that selects dynamically both the system to which a routing request is to be sent and the transaction's remote name. The alternative to using this program is to make these selections when a remote transaction is defined to CICS (static routing).

dynamic transaction routing (DTR). The automatic routing of a transaction, at the time it is initiated, from a transaction-owning region (TOR) to a suitable application-owning region (AOR).

dynamic storage area (DSA). Areas of storage (in the CICS region) that are used as needed by transactions being executed in the system. In CICS Transaction Server, there are five dynamic storage areas, CICS and user areas below the 16MB line, and CICS, user, and extended readonly areas above the 16MB line. CICS DSAs are preallocated at system initialization as specified by a series of system initialization parameters, CDSASZE, UDSASZE, ECDSASZE, ERDSASZE, and EUDSASZE.

dynamic transaction routing (DTR). Routing terminal data to an alternative transaction at the time the transaction is invoked. To do this, CICS allows the dynamic transaction routing program (DTRP) to intercept the terminal data and redirect it to any system and transaction it chooses. See also dynamic transaction routing program (DFHDYP).

dynamic transaction routing program (DFHDYP). A user-replaceable CICS program that selects dynamically both the system to which a transaction routing request is to be sent and the transaction's remote name. The alternative to using this program is to make these selections when a remote transaction is defined to CICS (static transaction routing). For programming information, see the Customization Guide.

E

EBCDIC. See Extended Binary-Coded Decimal Interchange Code.

ECB. See event control block (ECB).

EDF. See execution diagnostic facility (EDF).

EIB. See EXEC interface block (EIB).

ELPA. See extended link pack area (ELPA).

emergency restart. The CICS backout facility for an automatic restart following a system failure. It restores the recoverable resources updated by each interrupted transaction to the condition they were in when the transaction started.

EMP. See event monitoring point (EMP).

e**nd-of-day statistics.** In CICS Transaction Server, CICS statistics written to an SMF data set at the quiesce or termination of a CICS run, or at a user-specified time. In CICS/VSE, CICS statistics written to the CSSL transient data destination at the quiesce or termination of a CICS run, or at a user-specified time. End-of-day statistics are reset by an end-of-day statistics collection.

entry-sequenced data set (ESDS). A VSAM data set whose records are physically in the same order in which they were put in the data set. It is processed by addressed direct access or addressed sequential access and has no index. New records are added at the end of the data set.

event control block (ECB). An MVS or VSE control block that represents the status of an event. CICS task control uses ECBs.

event monitoring point (EMP). Point in the CICS code at which CICS monitoring data is collected. There are two types of EMP:

1. System-defined EMP, which collects predetermined CICS monitoring information and which cannot be relocated
2. User-defined EMP, which collects task monitoring information. See monitoring control table.

exception class data. CICS monitoring information on exception conditions raised by a transaction, such as queuing for VSAM strings or waiting for temporary storage. This data highlights possible problems in system operations. In CICS Transaction Server, monitoring of exception conditions (that is, the collection of exception class data) is activated by the MNEXC system initialization parameter. See MNEXC. In CICS/VSE, monitoring of exception conditions (that is, the collection of exception class data) is activated by the MONITOR=EXC system initialization parameter. See MONITOR.

exception trace entry. An entry made to the internal trace table and any other active trace destinations when CICS detects an exception condition. It gives information about what was happening at the time the failure occurred and what was being used.

EXCI. See external CICS interface (EXCI).

exclusive control. A type of access control in which VSAM keeps control of the control interval (CI) containing a specific record until a REWRITE, UNLOCK, or DELETE command is issued for that record. The purpose of exclusive control is to protect against simultaneous update.

EXEC interface. The high-level programming interface that uses CICS commands beginning with the verb EXEC. Synonymous with command-level interface and application programming interface.

EXEC interface block (EIB). A control block associated with each task in a CICS command-level environment. The EIB contains information that is useful during the execution of an application program (such as the transaction identifiers) and information that is helpful when a dump is being used to debug a program.

EXEC interface stub. The stub link-edited with every command-level program. It is part of the CALL interface between EXEC CICS commands and the CICS EXEC interface program (EIP).

execution diagnostic facility (EDF). A facility used for testing application programs interactively online, without making any modifications to the source program or to the program preparation procedure. The facility intercepts execution of the program at various points and displays information about the program at these points. Also displayed are any screens sent by the user program, so that the programmer can converse with the application program during testing just as a user would do on the production system.

execution interface program (EIP). Converts high-level (command-level) requests into the corresponding internal macro-level requests.

exit. An exit from and return to a CICS module at a stated functional point. The user can insert code at these points to enhance the program. See global user exit and task-related user exit.

exit programming interface (XPI). (CICS Transaction Server only.) Provides global user exit programs with access to some CICS services. It consists of a set of function calls that can be used in user exit programs to extend CICS functions.

extended addressing. The use of 31-bit addresses (above the 16MB line) which multiplies by 27 the range of virtual storage that can be addressed.

Extended Binary-Coded Decimal Interchange Code (EBCDIC). An interchange code in which the code pages consist of 8-bit coded characters.

extended link pack area (ELPA). A major element of MVS/ESA virtual storage above the 16MB line. It duplicates the link pack area (LPA). See the CICS Transaction Server Performance Guide for more information. See also extended addressing.

extended private area. An element of MVS/ESA virtual storage above the 16MB line. This area duplicates the private area except for the 16KB system region area. See the CICS Transaction Server Performance Guide for more information.

Extended Recovery Facility (XRF). A facility that increases the availability of CICS transaction processing, as seen by the users. Availability is improved by having a second CICS system (the alternate system) ready to continue processing the workload, if and when particular failures that disrupt user services occur on the first system (the active system).

extended system queue area (ESQA). A major element of MVS/ESA virtual storage above the 16MB line. This storage area contains tables and queues relating to the entire system. It duplicates above the 16MB line the system queue area (SQA). See the CICS Transaction Server Performance Guide for more information.

extent. Continuous space on a disk or diskette occupied by or reserved for a particular file or VSAM data space.

external call interface (ECI). An application programming interface that allows a non-CICS program running on a client to call a CICS program located on a CICS server.

external CICS interface (EXCI). A CICS application programming interface that helps to make CICS applications more easily accessible from non-CICS environments. It enables a non-CICS program (a client program) running in MVS to call a program (a server program) running in a CICS Transaction Server region and to pass and receive data by means of a communications area. The CICS program is invoked as if linked-to by another CICS program. For programming information about EXCI, see the External CICS Interface User's Guide.

external presentation interface (EPI). An application programming interface that allows a non-CICS client program to appear to a CICS server as one or more standard 3270 terminals. This enables the client to access, for example, CICS on System/390 transactions written for 3270 terminals, without needing to change the System/390 code.

external response time. Elapsed time from pressing the ENTER key or another AID key until the action requested by the terminal user is completed, and the next entry can be started. Elapsed time between the end of an enquiry or demand on a computer system and the beginning of the response.

extrapartition transient data. A CICS facility for temporarily saving data in the form of queues, called destinations. Each extrapartition TD destination requires a resource definition that links it to a SAM or QSAM data set outside the CICS region. Extrapartition destinations are used for data that is either coming from a source outside the region, or being directed from a source within the region to a destination outside the region. Extrapartition data written by CICS is usually intended for subsequent input to non-CICS batch programs. Examples of data that might be written to extrapartition destinations include logging records, statistics, and transaction error messages. Contrast with intrapartition transient data.

F

FEPI. See front end programming interface (FEPI).

file control table (FCT). Table containing the characteristics of the files accessed by file control.

Each entry in the FCT specifies the types of services to be allowed for a file, indicates the kind of access method used to get or put a record, and describes the record. Included as an appendage to each non-VSAM FCT entry, is the file definition.

file-owning region (FOR). Deprecated term for data-owning region (DOR), a CICS address space whose primary purpose is to manage files and databases. See also application-owning region (AOR), and terminal-owning region (TOR).

FILEA. Sample VSAM file provided for use by the CICS sample command-level applications.

first failure data capture (FFDC). A facility that provides the ability to capture the data relevant to a CICS exception condition as soon as possible after the condition has been detected.

FMID. In SMP, a keyword identifying the product (for example, CICS), release, and option to which a SYSMOD is applicable.

FOR. See file-owning region (FOR).

forward recovery. The process of restoring a backup copy and bringing it up to date by reapplying changes made to the file since the backup was taken. To facilitate forward recovery, CICS records after-images of file and database changes on the system log.

Some types of data base failure cannot be corrected by backward recovery; for example, failures that cause physical damage to the data base. Recovery from failures of this type is usually based on the following broad principles:

- Record, on the system log, an 'after-image' of every change.
- Take a backup, or 'image', copy of the data base at regular intervals.
- After the failure, use the information recorded on the system log to bring the backup copy to the most 'up-to-date' condition possible.

These operations are known as "forward recovery".

front end programming interface (FEPI). A separately-installable function of CICS Transaction Server that enables communication with non-LU6.2 partners by simulating an LU0 or LU2 device. FEPI allows CICS to communicate with existing applications on LU0 or LU2 systems without change to those applications. For details of FEPI, see the CICS Transaction Server 3.3 Front End Programming Interface General Information manual and the CICS Transaction Server 3.3 Front End Programming Interface User's Guide.

front-end transaction. In synchronous transaction-to-transaction communication, the transaction that acquires the session to a remote system and initiates a transaction on that system. Contrast with back-end transaction. For programming information, see the Application Programming Reference manual.

full trace. Option for formatting CICS trace entries. Full trace shows all the data for each trace entry. Compare with abbreviated trace. See the CICS Transaction Server Problem Determination Guide for more information.

function shipping. The process, transparent to the application program, by which CICS accesses resources when those resources are actually held on another CICS system. For further information, see the Intercommunication Guide.

function SYSMOD. An IBM product that can be installed with SMP/E. CICS Transaction Server is packaged as a function SYSMOD on a distribution tape. This contains distribution libraries and JCLIN data which SMP/E uses to create the target libraries.

G

generalized trace facility (GTF). In MVS, a trace data-collection routine. GTF traces the following system events: seek addresses on START I/O records, SRM activity, page faults, I/O activity, and supervisor services. Execution options specify the system events to be traced. CICS Transaction Server can use GTF as a destination for trace data. See GTFTR.

generic profile. In RACF, a profile that can provide protection for one or more resources. The resources protected by a generic profile have similar names and identical security requirements. For example, a generic data set profile can protect one or more data sets. Contrast with discrete profile.

global catalog. A system data set in which CICS records CICS system information. See also local catalog.

global user exit. A point in a CICS module at which CICS can pass control to a user-written program (known as an exit program), and then resume control when the program has finished. When an exit program is enabled for a particular exit point, the program is called every time the exit point is reached. See also task-related user exit (TRUE).

GTF. See generalized trace facility (GTF). See hierarchic indexed direct access method (HIDAMH).

H

hierarchical file system (HFS). The file system used by OS/390 UNIX System Services, in which files and directories are organized in a tree structure.

high performance option (HPO). System initialization parameter used to indicate whether CICS is to use the VTAM authorized path feature of the high performance option. The default is NO. You can code this parameter only in the system initialization table. See the CICS Transaction Server System Definition Guide for more information.

high private area. Part of the CICS address space, consisting of the local system queue area (LSQA), the scheduler work area (SWA), and subpools 229 and 230. The area at the high end of the CICS address space is not specifically used by CICS, but contains information and control blocks that are needed by the operating system to support the region and its requirements. See the Performance Guide for more information.

hiperspace™. A high-performance storage area in the processor or multiprocessor.

I

IBM CICSPlex System Manager for MVS/ESA (CICSPlex SM). An IBM CICS system-management product that provides a single-system image and a single point of control for one or more CICSPlexes that can be installed on heterogeneous operating systems.

I/O. See input/output (I/O).

IICE. See interval control element (ICE).

ICF. See integrated catalog facility (ICF).

IDCAMS. An MVS/DFP Access Method Services facility that allows users to define, manipulate, or delete VSAM data sets, define and manipulate integrated catalog facility catalogs, and copy, print, or convert SAM and ISAM data sets to VSAM data sets.

IIOP. See Internet Inter-ORB protocol.

image copy. A backup copy of a data set, used to restore the data set if necessary after a failure.

immediate shutdown. A shutdown of CICS in which tasks in progress are not allowed to complete normally. This form of shutdown is requested from the master terminal.

IMS. See Information Management System (IMS).

in-doubt. In CICS, the state at a particular point in a distributed UOW for which a two-phase commit syncpoint is in progress. The distributed UOW is said to be in-doubt when:

A subordinate recovery manager (or transaction manager) has replied (voted) in response to a PREPARE request, and

Has written a log record of its response to signify that it has entered the in-doubt state, and

Does not yet know the decision of its coordinator (to commit or to back out).

in-doubt window. The period between the sending of a syncpoint request to a remote system and the receiving of a reply. During this period, the local system does not know whether or not the remote system has committed its changes. If processing fails in the in-doubt window, recovery processing must resolve the status of any work that is in-doubt.

in-flight. A unit of work that is being processed when a system failure occurs.

in-flight task. A task that is in progress when a CICS system failure or immediate shutdown occurs. During emergency restart, a task that caused records to be written to the system log, but for which no syncpoint record has been found for the current LUW. This task was interrupted before the LUW completed.

index. In VSAM, an ordered collection of entries used to sequence and locate the records of a KSDS. Each entry consists of a key and a pointer. In ISAM, a table used to locate records in an ISAM data set.

index record. A system-created collection of VSAM index entries that are in collating sequence by the key in each of the entries.

indirect destination. A transient data destination that points to another destination, rather than directly to a queue. Indirect destinations allow you to refer to a single real destination by more than one name. For example, different message types can be associated with different indirect destinations. Changes to the indirect destination definitions can then cause all messages to go to the same real queue or to different queues as required.

Indirect destination entries address the TDQUEUE definition for the destination to which they indirectly refer (the target destination named in the INDIRECTNAME option of the TDQUEUE definition). The target destination can be an intrapartition, extrapartition, or remote destination, or another indirect destination.

initial program load (IPL). The process of loading system programs (for example, operating systems, CICS, DL/I, IMS, or SQL/DS), and preparing a system to run applications.

input/output (I/O). Pertaining to the movement of data between a processor and a peripheral device, to the functional unit or channel involved, and to the associated data.

installation. A particular computing system, in terms of the work it does and the people who manage it, operate it, apply it to problems, service it, and use the work it produces. The task of making a program ready to do useful work. This task includes generating a program, initializing it, and applying any changes to it.

installation verification procedure (IVP). A program or programs that are run at the end of installation of an IBM licensed program, in order to verify that the program is working correctly.

integrated catalog facility (ICF). A component that provides integrated catalog facility catalogs. See integrated catalog facility catalog.

integrated catalog facility catalog. A catalog that consists of a basic catalog structure, which contains information about VSAM and non-VSAM data sets, and at least one VSAM volume data set, which contains data about VSAM data sets only.

integrity. The quality of data that exists as long as destruction, alteration, loss of consistency, or loss of data are prevented.

Internet Inter-ORB protocol. An industry standard that defines formats and protocols to provide client/server semantics for distributed object-oriented application programs in a TCP/IP network. It is part of the Common Object Request Broker Architecture (CORBA) specification.

Interactive Problem Control System (IPCS). An MVS component that provides online problem diagnosis and reporting. A CICS IPCS exit enables the formatting of an MVS system dump.

interactive system productivity facility (ISPF). MVS interactive facility that can be used for CICS system administration tasks. Sample uses include defining RACF profiles, accessing SLR reports, and renaming a CICS SVC module or MVS system nucleus.

intercommunication. In CICS, a term embracing intersystem communication (ISC) and multiregion operation (MRO).

internal response time. Elapsed time from the message to start a transaction being received by CICS until the time that the transaction ends.

internal throughput rate (ITR). The number of completed transactions per processor-busy second. (Processor busy seconds can be calculated by multiplying elapsed seconds by the processor utilization percentage).

internal trace. A CICS trace facility that is present in virtual storage. When CICS detects an exception condition, an entry always goes to the internal trace table, even if you have turned tracing off. The internal trace table is a wraparound table whose size can be set by the TRTABSZ system initialization parameter in CICS Transaction Server (the TRACE system initialization parameter in CICS/VSE) and can be changed by the CICS SET TRACEDEST (CICS Transaction Server only) command. See the Problem Determination Guide for more information.

Internet Inter-ORB protocol (IIOP). The protocol used for transmitting CORBA messages across TCP/IP between possibly different ORB implementations.

interregion communication (IRC). The method by which CICS provides communication between a CICS region and another region in the same processor. Used for multiregion operation (MRO). Compare with intersystem communication.

intersystem communication (ISC). Communication between separate systems by means of SNA networking facilities or by means of the application-to-application facilities of VTAM. ISC links CICS systems and other systems, and may be used for communication between user applications, or for transparently executing CICS functions on a remote CICS system. Compare with multiregion operation and interregion communication.

interval control element (ICE). An element created for each time-dependent request received by the interval control program. These ICEs are logically chained to the CSA in expiration time-of-day sequence.

Expiration of a time-ordered request is detected by the expired request logic of the interval control program running as a CICS system task whenever the task dispatcher gains control. The type of service represented by the expired ICE is initiated, provided all resources required for the service are available, and the ICE is removed from the chain. If the resources are not available, the ICE remains on the chain and another attempt to initiate the requested service is made the next time the task dispatcher gains control.

interval control program (ICP). The CICS program that provides time-dependent facilities. Together with task control, interval control (sometimes called time management) provides various optional task functions (such as system stall detection, runaway task control, and task synchronization) based on specified intervals of time, or the time of day.

interval statistics. In CICS Transaction Server only, CICS statistics gathered at user-specified intervals and written to the SMF data set. See also end-of-day statistics, requested statistics, requested reset statistics, and unsolicited statistics.

intrapartition transient data (TD). A CICS facility for temporarily saving data in the form of queues, called destinations. All intrapartition TD destinations are held as queues in the same VSAM data set, which is managed by CICS. Data is written to the queue by a user task. The queue can be used subsequently as input data by other tasks within the CICS region. All access is sequential, governed by read and write pointers. Once a record has been read it cannot be read subsequently by another task. An intrapartition destination requires a resource definition containing information that locates the queue in the intrapartition data set. Applications that might use intrapartition queues include message switching, data collection, and queuing of orders.

IPCS. See Interactive Problem Control System (IPCS).

IRC. See interregion communication (IRC).

ISPF. See interactive system productivity facility (ISPF).

IVP. See installation verification procedure (IVP).

J

Java. An interpreted object-oriented language, similar to C++, which can be used to build programs that are platform independent in both source and object form. Its unique operational characteristics, which span Web browser and network computers as well as servers, enable new client/server functions in Internet applications while enforcing a discipline that enables software management across almost any hardware platform.

The Java language can be used to construct Java applets and Java applications. Both are used in the CICS Transaction Gateway.

Java virtual machine. A software implementation of a central processing unit (CPU) that runs compiled Java code.

JCL. See job control language (JCL).

JES. See job entry subsystem (JES, JES2, JES3).

job control language (JCL). Control language used to describe a job and its requirements to an operating system.

job entry subsystem (JES, JES2, JES3). The subsystem used in CICS with XRF to route commands and queries from the alternate system to the active system.

journaling. The recording of information onto any journal (including the system log), for possible subsequent processing by the user. The primary purpose of journaling is to enable forward recovery of data sets. A data set can be reconstructed by applying transactions in the journal against a previous version of the data set. Journaling can also be used for any other user-defined purpose, such as auditing, accounting, or performance analysis.

The above definition of journaling excludes logging. The term journaling is sometimes used to include both logging and journaling as above defined. See logging.

JVM. See Java virtual machine.

K

kernel domain. Major component of CICS providing a consistent linkage and recovery environment for CICS. The application programmer has no external interface to kernel linkage.

Kernel Linkage. A component of CICSPlex SM that is responsible for building data structures and managing the interfaces between the other CICSPlex SM components. The environment built by Kernel Linkage is known as the method call environment.

key. (ISO) One or more characters within a set of data that contains information about that set, including its identification. The field in a segment used to store segment occurrences in sequential order. A field used to search for a database segment or a data set record.

key 0. MVS storage with storage key 0. Key 0 storage is non-fetch protected. By default, CICS uses key 0 storage for the extended read-only dynamic storage area (ERDSA). If the system initialization parameters include RENTPGM=NOPROTECT, the ERDSA is placed in CICS-key storage, MVS storage key 8.

key-sequenced data set (KSDS). A VSAM file whose records are loaded in key sequence and controlled by an index.

keypoint. The periodic recording of system information and control blocks on the system log-also the data so recorded. See also activity keypoint, and warm keypoint.

KSDS. See key-sequenced data set (KSDS).

L

Language Environment/390. A runtime library that establishes a common execution environment for a number of SAA languages. See also Systems Application Architecture (SAA).

library. In CICS manuals, often used as a synonym for program library. In MVS, a synonym for a partitioned data set.

library lookaside (LLA). A facility in MVS/ESA that reduces library I/O activity by keeping selected directory entries in storage, instead of making repetitive searches of DASD.

link pack area (LPA). A major element of MVS/ESA virtual storage below the 16MB line. The storage areas that make up the LPA contain all the common reentrant modules shared by the system. The LPA provides economy of real storage by sharing one copy of the modules, protection because LPA code cannot be overwritten even by key 0 programs, and reduced pathlength because the modules can be branched to. The LPA is duplicated above the 16MB line as the extended link pack area (ELPA). See the CICS Transaction Server Performance Guide for more information.

linkage editor. A computer program used to create one load module from one or more independently-translated object modules or load modules by resolving cross references among the modules. All CICS application programs need to be processed by the linkage editor (link-edited) before execution. See source program, object module, load module, compiler, assembler.

LLA. See library lookaside (LLA).

loader domain. Major component of CICS used by the domains of the CICS system to obtain access to storage-resident copies of nucleus and application programs, maps, and tables. In order to provide this, the loader domain interfaces with MVS to perform loading of programs into CICS-managed storage (DSA/EDSA) and scanning of the MVS link pack area.

local catalog. A system data set that CICS uses to record data used by the internal workings of CICS. See also global catalog.

local shared resources (LSR). Files that share a common pool of buffers and a common pool of strings; that is, control blocks supporting I/O operations. Contrast with nonshared resources.

local system queue area (LSQA). An element of the CICS address space. It generally contains the control blocks for storage and contents supervision. See the CICS Transaction Server Performance Guide for more information. See also high private area.

local work area. Area provided for the use of a single task-related user exit program. It is associated with a single task and lasts for the duration of the task only.

log. A recording of changes made to a file. This recording can be used for subsequent recovery of the file. See also dynamic log, journal, and system log.

logging. The recording (by CICS) of recovery information onto the system log, for use during emergency restart. A specific journaling function that records changes made to the system activity environment and database environment. These records are required for recovery and backout support by CICS (and the user) following an abnormal termination.

logical partition (LP or LPAR). A partition, in a CEC, capable of running its own MVS or VSE image. An LP comprises a set of hardware resources (processors, storage, channels, and so on), sufficient to allow a system control program such as MVS or VSE to be executed.

logical unit (LU). In SNA, a port through which a user gains access to the services of a network.

logical unit of work (LUW). A sequence of processing actions (database changes, for example) that must be completed before any of the individual actions can be regarded as committed. When changes are committed (by successful completion of the LUW and recording of the syncpoint on the system log), they do not need to be backed out after a subsequent failure of the task or system. The end of an LUW is marked in a transaction by a syncpoint, issued either by the user program or by CICS at the end of task. In the absence of user syncpoints, the entire task is an LUW.

logon. The act of establishing a session with VTAM. Contrast with signon.

long running mirror. A mirror task that waits for the next syncpoint in a session, even though it logically does not need to do so (applicable only to MRO links).

LPA. See link pack area (LPA).

LSQA. See local system queue area (LSQA).

LSR. See local shared resources (LSR).

LU. See logical unit (LU).

LUW. See logical unit of work (LUW).

M

macro. An instruction that when executed causes the execution of a predefined sequence of instructions in the source language. The predefined sequence can be modified by parameters in the macro. CICS RDM macros are assembler macros and are converted by the assembler. Contrast with command.

main storage. (ISO) Program-addressable storage from which instructions and data can be loaded directly into registers for subsequent execution or processing. See also real storage, storage, virtual storage.

map. In BMS, a format established for a page or a portion of a page, or a set of screen format descriptions. A map relates program variables to the positions in which their values appear on a display device. A map contains other formatting information such as field attributes. A map describes constant fields and their position on the display, the format of input and output fields, the attributes of constant and variable fields, and the symbolic names of variable fields.

map set. In BMS, one or more maps combined in a map set. The effects of this combination are to reduce the number of entries in the PPT, and to load simultaneously all maps needed for one application.

master terminal. In CICS, the terminal at which a designated operator is signed on. In IMS, the logical terminal that has complete control of IMS resources during online operations.

mirror task. CICS task that services incoming requests that specify a CICS mirror transaction (CSMI, CSM1, CSM2, CSM3, CSM5, CPMI, CVMI, or a user-defined mirror transaction identifier). For more information, see the Resource Definition manual.

mirror transaction. CICS transaction that recreates a request that is function shipped from one system to another, issues the request on the second system, and passes the acquired data back to the first system.

MLPA. See modified link pack area (MLPA).

modegroup. A VTAM LOGMODE entry, which can specify (among other things) the class of service required for a group of APPC sessions.

modename. The name of a modeset and of the corresponding modegroup.

modeset. In CICS, a group of APPC sessions. A modeset is linked by its modename to a modegroup (VTAM LOGMODE entry) that defines the class of service for the modeset.

modified data tag (MDT). In the attribute byte of each field in a BMS map, a bit that determines whether the field should be transmitted on a READ MODIFIED command (the command used by CICS for all except copy operations).

modified link pack area (MLPA). An element of MVS/ESA virtual storage. This area provides a temporary extension to the PLPA existing only for the life of the current IPL. You can use this area to add or replace altered LPA-eligible modules without having to recreate the LPA. See also link pack area (LPA) and pageable link pack area (PLPA).

MRO. See multiregion operation (MRO).

multitasking. Concurrent execution of application programs within a CICS region.

multithreading. Use, by several transactions, of a single copy of an application program.

MVS image. (CICS Transaction Server only.) A single copy of the MVS operating system. Note that a single processing environment can support more than one MVS image.

N

NCP. See network control program (NCP).

NEP. See node error program (NEP).

NETNAME (netname). In CICS, the name by which a CICS terminal or a CICS system is known to VTAM.

network control program (NCP). A program that controls the operation of a communication controller (3745, 3725, 3720, 3705) in which it resides. NCP builds the backup sessions to the alternate CICS system for XRF-capable terminals. NCP is generated by the user from a library of modules.

node error program (NEP). A user-replaceable program used to allow user-dependent processing whenever a communication error is reported to CICS

nonconversational. A mode of CICS operation in which resources are allocated, used, and released immediately on completion of the task.

nonshared resources (NSR). Files with their own set of buffers and control blocks. Contrast with local shared resources (LSR).

nucleus. That portion of the CICS region that holds the CSA, management modules, control tables, and resident application programs.

O

open key storage. In MVS storage protection, storage with storage key 9. called key-9 storage. In open key storage, fetch and store operations are permitted, regardless of the access key. CICS user-key storage is in MVS open key storage.

P

pacing. In SNA, a technique by which a receiving component controls the rate of transmission of a sending component to prevent overrun or congestion.

page. A fixed-length block that has a virtual address and that is transferred as a unit between real storage and auxiliary storage. See paging. Information displayed at the same time on a display device.

page fault. A reference by an executing program to instructions or data that are not in real storage. When a page fault occurs, the page in virtual storage that contains the referenced data must be paged into real storage.

pageable link pack area (PLPA). An element of MVS/ESA virtual storage. This area contains supervisor call routines, access methods, and other read-only system programs along with read-only reenterable user programs selected by an installation to be shared among users of the system. Optional functions or devices selected by an installation during system generation add additional modules to the PLPA. See also link pack area (LPA) and modified link pack area (MLPA).

paging. The transfer of pages between real storage and auxiliary storage. The process of transferring pages between real storage and the external page storage known as the page data set.

parallel sysplex. An MVS sysplex where all the MVS images are linked through a coupling facility.

partitioned data set (PDS). In MVS, a data set in direct access storage that is divided into partitions called members. A member can contain a program or data. Program libraries are held in partitioned data sets.

PassTicket. In RACF, a PassTicket is a character string, generated by a program that can be used in place of a password, with the following constraints: a specific PassTicket may only be submitted for validation once; the PassTicket must be used within ten minutes of being generated. For more information, see the *CICS-RACF Security Guide*.

peer-to-peer. A form of distributed processing, in which the front-end and back-end of a conversation switch control between themselves. It is communication between equals.

persistent session. (1) A network management session in the NetView program that remains active even though there is no activity on the session for a specified period of time.

(2) An LU-LU session that VTAM retains following the failure of a VTAM application program. Following the application program's recovery, the application program either restores or terminates the session. For more information about how CICS uses VTAM persistent sessions, see the *Recovery and Restart Guide*, the *Intercommunication Guide*, or *VTAM Programming for MVS/ESA, VM/SP, and VM/ESA*, SC31-6436.

physical map. A set of instructions telling BMS how to format a display for a given device. BMS does this by imbedding control characters in the data stream.

pipe. A one-way communication path between a sending process and a receiving process. In an external CICS interface implementation, each pipe maps onto one MRO session, where the client program represents the sending process and the CICS server region represents the receiving process. For programming information about EXCI, see the External CICS Interface Guide.

PLPA. See pageable link pack area (PLPA).

pool thread. A thread which is used by the CICS DB2 attachment facility for transactions and commands that do not use an entry thread or a command thread.

PR/SM. See processor resource/systems manager (PR/SM).

prefixed save area (PSA). An element of MVS/ESA virtual storage which contains processor-dependent status information. See the CICS Transaction Server Performance Guide for more information.

prefixing. Specifying at system initialization that you want CICS to prefix the resource names that it passes to RACF for authorization with the RACF userid under which the CICS region is running.

preset terminal security. When a CICS region is started, the signing on of selected terminals as "users" whose userids are the terminal identifiers. Persons using these terminals have the authorizations given to the terminals.

primary index. In VSAM, the set of primary keys that provide the standard path for access to the data set.

primary key. In each record of a VSAM KSDS, an identifying field. The key of each record is a field in a predefined position within the record. Each key must be unique in the data set.

primary logical unit (PLU). In an SNA LU-LU session, the logical unit that issued the bind request that established the session. The PLU contains the primary half-session. The same logical unit can be the PLU in some sessions and the secondary logical unit (SLU) in others.

principal facility. The terminal or logical unit that is connected to a transaction at its initiation. Contrast with alternate facility.

processor resource/systems manager (PR/SM). The feature on a 3090 Processor Complex that offers flexible partitioning of a 3090 processing system into a number of logical partitions. Each partition within a PR/SM environment supports its own MVS or VSE image and VTAM, resulting in a multi-MVS or VSE environment.

profile. In CICS, a set of options specified in a resource definition that can be invoked by a transaction definition. Profiles control the interactions between the transaction and terminals or logical units. CICS supplies profile definitions suitable for most purposes. If a transaction definition does not specify a profile, a standard profile is used. In CICS Transaction Server only, in RACF, data that describes the significant characteristics of a user, a resource, a group of users, or a group of resources. See resource profile, discrete profile, generic profile, user profile, resource group profile, data set profile.

program check. A condition that occurs when programming errors are detected by a processor during execution.

program compression. An operation performed by program control to relieve space in the DSA during a short-on-storage condition. The PPT (PD) is searched to identify programs that have been dynamically loaded and are currently not in use. If a program is not in use, the space it occupied is reclaimed.

program error program (PEP). A user-replaceable program containing code to obtain program addressability, access the COMMAREA, and return control to the CICS abnormal condition program (DFHACP) through an EXEC CICS RETURN command. For programming information, see the Customization Guide.

program list table (PLT). CICS control table containing a list of programs. The programs in a PLT can be executed as a group during CICS startup or shutdown, and can be enabled and disabled as a group by a single CEMT transaction. See PLTPI, PLTSD.

program status word (PSW). Area in MVS or VSE storage that holds the status of the system as a program executes. In a formatted dump produced by a program check or an abend, the PSW and the registers at the time of the program check or abend are printed at the beginning of the dump. These help to identify the program that was executing, and the offset within that program at which the problem occurred.

program temporary fix (PTF). A temporary solution or bypass of a problem diagnosed by IBM field engineering as the result of a defect in a current, unaltered release of the program. See authorized program analysis report (APAR).

PSA. See prefixed save area (PSA).

pseudoconversational. A type of CICS application design that appears to the user as a continuous conversation, but that consists internally of multiple tasks -- also called "transaction-oriented programming."

Q

QSAM. See queued sequential access method (QSAM).

quasi-reentrant. The attribute used to describe CICS application programs that run under the CICS quasi-reentrant task control block (QR TCB). This means that:

CICS obtains a separate copy of program working storage for each task that executes application program code.

CICS allows only one task at a time to execute application program code. In this way, CICS ensures the necessary serialization of user application programs that access any kind of shared resources, whether CICS- or user-managed. This means that different tasks cannot interfere with each other. Thus the user application program need not be reenterable strictly according to the DFSMS program management definition. See Reentrant.

Although only one user task can execute an application program at any one time, a second user task can enter the program before another task has finished with it, unlike a serially reusable module as defined by DFSMS. This is because user applications programs give up control part way through execution whenever they issue an EXEC CICS command that causes a wait. Thus a user program can be in use concurrently by more than one task, indicated by the use count maintained by CICS, which can be greater than one. Whenever an application program receives control, it should be in the same state as when it relinquished control on a previous invocation.

queued sequential access method (QSAM). An extended version of BSAM that incorporates queues of input and output blocks that are awaiting processing and transfer respectively.

R

RACF. See Resource Access Control Facility (RACF).

RAIA. See receive-any input area (RAIA).

RDO. See resource definition online (RDO).

RDSA. See read-only dynamic storage area (RDSA)

read integrity. An attribute of a read request, which ensures the integrity of the data passed to a program that issues a read-only request. Contrast with dirty read.

read-only dynamic storage area (RDSA). The key-0 storage area for all reentrant programs and tables below the 16MB boundary.

real storage. The main storage in a virtual storage system. Physically, real storage and main storage are identical. Conceptually, however, real storage represents only part of the range of addresses available to the user of a virtual storage system.

receive-any control element (RACE). Type of control field held in the CICS receive-any pool set aside for VTAM receive-any operations. The number of RACEs maintained depends on the RAPOOL and MXT system initialization parameters and on the number of active tasks. See the CICS Transaction Server System Definition Guide or the CICS/VSE System Definition and Operations Guide for more information.

receive-any input area (RAIA). Type of input area held in the CICS receive-any pool set aside for VTAM receive-any operations. The number of RACEs maintained depends on the RAPOOL and MXT system initialization parameters and on the number of active tasks. See the CICS Transaction Server System Definition Guide or the CICS/VSE System Definition and Operations Guide for more information.

recoverable resource. A resource whose definition specifies that CICS is to take measures to ensure the resource's integrity.

recovery. The process of returning the system to a state from which operation can be resumed. The restoration of resources following an error.

reentrant. (From the MVS Assembler Services Guide). The attribute that describes a load module, of which only one copy is loaded into virtual storage to satisfy the requirements of any number of tasks. A single copy of a reentrant load module can be executed concurrently by any number of tasks. A reentrant load module is also one that does not modify itself, and must be link-edited with the RENT attribute.

region. In MVS, a variable-size subdivision of virtual storage that is allocated to a job step or system task. CICS Transaction Server runs in an MVS/ESA region, usually referred to as the CICS region.

relative record data set (RRDS). A VSAM data set organization, in which records are of fixed length and are accessed by their relative record numbers. The relative record number (RRN) of a record is its displacement (in records) from the beginning of the data set.

relay program. In transaction routing, a CICS program that provides the communication mechanism between a locally-connected terminal and a transaction in a remote system. The relay program is invoked by the relay transaction.

relay transaction. In transaction routing, a CICS transaction that handles communication between a locally-connected terminal and a transaction in a remote system. The relay transaction invokes the relay program.

remote. In data communication, pertaining to devices that are connected to a data processing system through a data link. Synonym of link-attached. Contrast with local.

Remote Method Invocation (RMI). A protocol that allows objects to be distributed over the network; that is, a Java program running on one computer can call the methods of an object running on another computer.

remote resource. In CICS intercommunication, a resource that is owned by a remote system. Contrast with local resource.

remote system. In CICS intercommunication, a system that the local CICS system accesses via intersystem communication or multiregion operation. Contrast with local system.

request parameter list (RPL). In VTAM, a control block that contains the parameters necessary for processing a request for data transfer, for connecting or disconnecting a terminal, or for some other operation.

request/response unit (RU). In SNA, the basic unit of information entering or leaving the transmission subsystem. It may contain data, acknowledgments, control commands, or responses to commands.

resource. Any facility of the computing system or operating system required by a job or task, and including main storage, input/output devices, the processing unit, data sets, and control or processing programs.

Resource Access Control Facility (RACF). (CICS Transaction Server only.) An IBM licensed program that provides for access control by identifying and verifying users to the system, authorizing access to protected resources, logging detected unauthorized attempts to enter the system, and logging detected accesses to protected resources.

r**esource definition online (RDO).** The primary method of defining resources to CICS. Resource definitions are created interactively with the CEDA transaction, or by using the utility DFHCSDUP. Both methods store definition in the CICS system definition data set (CSD). At CICS initialization, CSD definitions are selectively installed as CICS system tables, controlled by a user-supplied list of definitions. CEDA-defined resource definitions can be installed while CICS is active and used immediately.

resource manager interface (RMI). A program or a group of programs that you write to enable you to structure calls from your CICS system in such a way that they can access non-CICS resources, such as databases, that you would not normally be able to access. An RMI is written using the CICS task-related user exit interface. DBCTL, for example, is accessed by means of an RMI. See also task-related user exit.

resource measurement facility (RMF). (CICS Transaction Server only.) An IBM licensed program that collects system-wide data describing the processor activity (WAIT time), I/O activity (channel and device utilization), main storage activity (demand and swap paging statistics), and system resources manager (SRM) activity (workload). RMF produces two types of report, system-wide reports and address-space reports.

resource security. In CICS Transaction Server, the facility provided by CICS and RACF for the control of access to resources protected by RACF security classes. The resources that can be protected include transactions, data sets, and transient data destinations.

RETAIN®. Database used by IBM Support Centers to record all known problems with IBM licensed programs.

RLDS. See recovery log data set (RLDS).

RMF. See resource measurement facility (RMF).

RMI. See resource manager interface (RMI) and remote method invocation.

RMODE. In MVS and VSE, an attribute that specifies the residence mode of the load module produced by the linkage editor. The possible values are RMODE(24) and RMODE(ANY). A program link-edited with RMODE(24) must reside below the 16MB line. If a program is link-edited with RMODE(ANY), CICS loads it above the 16MB line if possible.

rollback. A programmed return to a prior checkpoint. In CICS, the cancellation by an application program of the changes it has made to all recoverable resources during the current logical unit of work.

RPL. See request parameter list (RPL).

RRDS. See relative record data set (RRDS).

runaway task. A task that has been dispatched and does not return control to CICS within a user-specified time interval. The program being used by this task is in a loop between two CICS requests. The task control program abends the task after expiration of this time interval, which is called the runaway task time interval.

S

SAA. See Systems Application Architecture (SAA).

SAA AD/Cycle COBOL/370™. The compiler that supports AD/Cycle (Language Environment/370).

SAA communications interface. A programming interface that allows program-to-program communication using the SNA APPC protocols. See also Systems Application Architecture (SAA).

SAF. See System Authorization Facility (SAF).

sample statistics program (DFH0STAT). Batch program supplied with CICS which provides information that is useful in calculating the storage requirements of a CICS Transaction Server system, for example, the sizes of the dynamic storage areas.

scheduler work area (SWA). An element of the CICS address space. The SWA is made up of subpools 236 and 237 which contain information about the job and the step itself. Almost anything that appears in the job stream for the step creates some kind of control block in this area. See the CICS Transaction Server Performance Guide for more information.

Screen Definition Facility (SDF). An interactive tool used to define and maintain maps, map sets, and partition sets for CICS and BMS applications. For CICS/MVS only, the relevant version of SDF is Screen Definition Facility II (SDF II), program number 5665-366. For CICS/VSE, it is Screen Definition Facility/CICS VSE (SDF/CICS VSE), release 1.5, program number 5746-XXT.

SDUMP. See system dump (SDUMP).

SDWA. System diagnostic work area.

secondary index. In IMS or VSAM, any index used to provide a path for access to a data set other than that provided by the primary keys. See alternate index.

secondary logical unit (SLU). In an SNA session, the logical unit that received the bind request that established the session. The same logical unit can be the SLU in some sessions and the primary logical unit (SLU) in others.

serially reusable. The attribute that describes a serially reusable load module. Only one copy of a serially reusable load module is loaded into virtual storage to satisfy the requirements of any number of tasks, but only one task can execute the module at any one time. If the copy is in use when a request is issued for the module, the task requiring the module is placed in a wait condition until the module is available.

(From the DFSMS Program Management manual). The module is designed to be reused and therefore must contain the necessary logic to reset control variables and data areas at entry or exit. A second task may not enter the module until the first task has finished.

Service Level Reporter (SLR). An IBM licensed program that produces reports on CICS performance and service levels. The reports can be used for performance management and many other purposes.

service policy. A set of performance goals for all MVS images using MVS workload management in a sysplex. There can be only one active service policy for a sysplex, and all subsystems in goal mode within that sysplex process towards that policy. However, you can create several service policies, and switch between them to cater for the different needs of different processing periods.

shared dynamic storage area (SDSA). The user-key storage area for any non-reentrant user-key RMODE(24) programs, and also for any storage obtained by programs issuing EXEC CICS GETMAIN commands for storage below the 16MB boundary with the SHARED option. For more details, see the Recovery and Restart Guide, and the Performance Guide.

shippable terminal. In transaction routing, a terminal whose definition can be shipped to another CICS system when the other system requires a remote definition of that terminal.

short-on-storage (SOS). The condition in CICS that occurs when requests for storage from the dynamic storage areas exceed available storage. CICS cannot satisfy these requests, or can satisfy them only by using some of the storage cushion, even when all programs that are eligible for deletion, and are not in use, have been deleted. See also storage cushion and program compression.

shunted. The status of a UOW that has failed at one of the following points:

- While in-doubt during a two-phase commit process
- While attempting to commit changes to resources at the end of the UOW
- While attempting to back out the UOW

If a UOW fails for one of these reasons, it is removed (shunted) from the primary system log (DFHLOG) to the secondary system log (DFHSHUNT) pending recovery from the failure.

signon. In CICS, to perform user identification and verification. The CICS user signs on to CICS using a CICS transaction: CESN in CICS Transaction Server, CSSN in CICS/VSE. Contrast with logon, which means to establish a session with VTAM.

SIGNOFF

(CICS Transaction Server only.) EXEC CICS command used to sign off the terminal or principal facility that you previously signed on. For programming information, see the CICS Transaction Server Application Programming Reference manual.

single session. A type of APPC connection with limited function. A single-session connection supports only one session and does not have SNA service manager support.

single threading. The execution of a program to completion. Processing of one transaction is completed before another transaction is started. Compare with multithreading.

SIT. See system initialization table (SIT). System initialization parameter that specifies the suffix, if any, of the system initialization table (SIT) that you want CICS to load at the start of initialization. If you omit this parameter, CICS loads the pregenerated, default SIT (DFHSIT$$ in CICS Transaction Server, DFHSIT in CICS/VSE).

SLR. See Service Level Reporter (SLR).

SLU. See secondary logical unit (SLU).

SMF. See system management facility (SMF).

SMP/E. See System Modification Program Extended (SMP/E).

SMSVSAM. The name of the VSAM server that provides VSAM record-level sharing (RLS). See also VSAM RLS.

SNA. See Systems Network Architecture (SNA).

snap dump. A dump that can be requested by a task at any time during which that task is being processed. Also known as snapshot dump.

SOS. See short-on-storage (SOS).

SP commands. The subset of CICS API commands (COLLECT, DISCARD, INQUIRE, PERFORM, and SET) that require the special CICS translator option, SP, and for which command security checking can be done. For programming information, see the CICS Transaction Server System Programming Reference manual.

spanned record. In a VSAM KSDS or ESDS, a logical record that occupies more than one control interval.

SRM. See system resources manager (SRM).

SSA. See segment search argument (SSA).

started transaction. A CICS transaction initiated by a terminal user can start other transactions by means of a CICS START command. A transaction started in this way is known as a started transaction.

statistics. System statistics are accumulated continually by CICS management programs in CICS system tables during the execution of CICS. System statistics can be captured and recorded, either on request or automatically at intervals, by any operator whose security code allows access to such information. In addition, system statistics are recorded on shutdown of the system.

See unsolicited statistics, end-of-day statistics, requested statistics, and requested reset statistics.

storage accounting area (SAA). A field at the start of a CICS storage area that describes the area and enables CICS to detect some storage violations. Each CICS storage area has either an SAA or a storage check zone.

storage check zone. A pair of fields at the beginning and end of a CICS storage area that enable CICS to detect some storage violations. Each CICS storage area has either a storage check zone or a storage accounting area (SSA).

storage cushion. A noncontiguous area of storage in the dynamic storage areas reserved for use by CICS when processing a short-on-storage condition.

storage key. A key associated with each 4KB block of storage that is available in the CICS region. Access to CICS storage is controlled by key-controlled storage protection. When key-controlled protection applies to a storage access, a store operation (write) is permitted only when the storage key matches the access key associated with the request; a fetch (read) is permitted when the keys match or when the fetch-protection bit of the storage key is zero. In most cases, the access key for a storage operation is the PSW key in the current PSW. For information about how MVS determines when access keys match storage keys, see the IBM Enterprise Systems Architecture/390 Principles of Operation manual.

storage protection. An optional facility in CICS/ESA 3.3 that enables users to protect CICS code and control blocks from being overwritten inadvertently by application programs.

storage violation. An error in a storage accounting chain in the dynamic storage area. A storage violation can be detected by the storage manager domain in CICS Transaction Server or the storage manager program in CICS/VSE.

stress. A shortage of free space in the DSA or EDSA, such that CICS cannot recover from virtual storage depletion.

string. A string of elements of the same nature, for example, character string or bit string.

stub. In CICS, two different types of object:

For the EXEC interface, the function-dependent module(s) associated with the EXEC interface nucleus DFHEIP. These stubs have names (such as DFHETC, DFHEFC) and are invoked by DFHEIP.

In various contexts, including task related user exits, a piece of code that is link-edited with an application program and serves the dual function of satisfying the CALL requirement for a target address, and finding the entry point of DFHEIP.

Both of these types of stub are part of the path between an application call and the functional management module that supports the request.

subspace group facility. A facility in MVS/ESA 5.1, which can be used for storage isolation to preserve data integrity within an address space. A subspace group is a group of subspaces and a single base space, where the base space is the normal MVS address space as in releases of MVS/ESA prior to MVS/ESA 5.1. The subspace group facility provides a partial mapping of the underlying base space, so that only specified areas of storage in the base space are exposed in a particular subspace. An application server is a program that manages multiple application programs running in a single address space. The subspace group facility prevents these application programs from overwriting each other. An authorized program can assign a unique section of address space private storage to each program running in the address space. The programs can reference only the storage assigned to them, which prevents them from accidentally overwriting each other's data and code. The subspace group facility is available in both sysplex and non-sysplex environments.

subtasking. The use by CICS of an additional TCB to perform certain functions, such as VSAM requests, as system subtasks. This is in addition to the TCB that CICS uses for normal processing.

supervisor call (SVC). An MVS instruction that interrupts the program being executed and passes control to the supervisor so that it can perform a specific service indicated by the instruction.

surrogate TCTTE. In CICS transaction routing, a TCTTE in the transaction-owning region that is used to represent the terminal that invoked, or was acquired by, the transaction. See surrogate terminal.

surrogate terminal. A terminal whose terminal definition is shipped from a terminal owning region (TOR). See surrogate TCTTE.

SWA. See scheduler work area (SWA).

sympathy sickness. In intercommunication, a condition in which the impaired performance of one region spreads to, and impairs the performance of, connected regions. For more details, see the Intercommunication Guide.

synchronization level (sync level). The level of synchronization (0, 1, or 2) established for an APPC session between intercommunicating CICS transactions. Level 0 gives no synchronization support, level 1 allows the exchange of private synchronization requests, and level 2 gives full CICS synchronization support with backout of all updates to recoverable resources if failure occurs.

syncpoint (sync point). A logical point in execution of an application program where the changes made to the databases by the program are consistent and complete and can be committed to the database. The output, which has been held up to that point, is sent to its destination(s), the input is removed from the message queues, and the database updates are made available to other applications. When a program terminates abnormally, CICS recovery and restart facilities do not backout updates prior to the last completed syncpoint.

SYSPLEX. A set of one or more MVS systems, where a system is a collection of data processing services under the control of a single control program.

System Authorization Facility (SAF). MVS facility through which CICS communicates with an external security manager (for example, RACF). You can use SAF customization options to customize the interface between CICS and the external security manager (ESM).

system dump (SDUMP). An MVS SDUMP, which can be formatted with a CICS IPCS exit to show all control blocks and storage areas in the CICS region. See the CICS Transaction Server Problem Determination Guide. A system dump can be requested with the EXEC CICS PERFORM DUMP command. See the CICS Transaction Server System Programming Reference manual. See system dump code.

system dump code. A name of up to eight characters by which a system dump will be known. A system dump code can be defined by CICS or by the user and identifies a set of system actions held in the form of an entry in the system dump table. For more information, see the Problem Determination Guide. The Messages and Codes manual contains a list of the CICS system dump codes. See also dump code.

system dump table (SDT). A CICS table which may contain an entry for each system dump code. See dump table and system dump table entry.

system initialization parameter. Parameter used to define capabilities of a CICS system at the time of system initialization. A system initialization parameter can be predefined in the system initialization table (SIT), or specified dynamically from the console, in the SYSIN data set, or as a parameter in the startup JCL.

system initialization program (DFHSIP SIP). CICS program that builds a CICS system using the resources you have defined and any user-designed or purchased applications. DFHSIP receives instructions from system initialization parameters.

system initialization table (SIT DFHSIT). A CICS table that contains information to initialize and control system functions, module suffixes for selection of user-specified versions of CICS modules and tables, and information used to control the initialization process. You can generate several SITs, using the resource definition macro DFHSIT, and then use the SIT system initialization parameter to select the one that best meets your current requirements at initialization time.

system log. A log stream maintained by CICS for back-out recovery purposes.

The system log is used by CICS to recover data to a consistent state following:

- The failure of an individual transaction
- The failure of a CICS region
- The failure of a connection with a partner in a distributed unit of work

User transactions are allowed to write their own recovery records to the system log for use in an emergency restart, but the system log cannot be used for forward recovery log or autojournal records.

Contrast with general log.

system logger. A central logging facility provided by MVS/ESA SP 5.2. The MVS system logger provides an integrated MVS logging facility that can be used by system and subsystem components. For example, it is used by the CICS log manager.

system management facility (SMF). MVS management program. CICS stores monitoring and statistical data on SMF data sets. See monitoring and statistics.

System Modification Program Extended (SMP/E). An IBM licensed program used to install software (for example, CICS Transaction Server) and software changes on MVS/ESA.

system queue area (SQA). A major element of MVS/ESA virtual storage below the 16MB line. This storage area contains tables and queues relating to the entire system. Its contents are highly dependent on the configuration and job requirements at installation. The equivalent area above the 16MB line is the extended system queue area (ESQA).

system recovery table (SRT). A table listing the ABEND or abnormal condition codes that CICS will intercept.

Systems Application Architecture (SAA). A set of common standards and procedures for working with IBM systems and data. SAA enables different software, hardware and network environments to coexist. It provides bases for designing and developing application programs that are consistent across different systems. See also: AD/Cycle Language Environment/370, CPI (common programming interface), SAA communications interface, common user access (CUA), and SAA resource recovery.

Systems Network Architecture (SNA). A description of logical structures, formats, protocols, and operational sequences for transmitting information units through, and controlling the configuration and operation of, networks. The structure of SNA allows the users to be independent of, and unaffected by, the specific facilities used for information exchange.

T

task. In CICS, a single instance of the execution of a transaction. Contrast with transaction. A unit of work for the processor; therefore the basic multiprogramming unit under the MVS or VSE control program.

task control area (TCA). An area of main storage acquired by CICS when a task is first dispatched. It is used to control the processing of the task. Once acquired, the TCA exists until the task is terminated. It contains the current status of the task, its relative dispatching priority, and parameters and information being passed between CICS and the application program. During execution of the task, the user can change the priority through task control services; further processing of the task is scheduled accordingly.

task control block (TCB). In CICS Transaction Server, an MVS control block. A TCB is created for each MVS task. Several TCBs are created for CICS management programs. All CICS application programs and all non-reentrant CICS code run under a single quasi-reentrant TCB. In CICS/VSE, a VSE control block. A TCB is created for each VSE task.

task-related user exit (TRUE). A task-related user exit enables you to write a user exit program that is associated with specified events in a particular task, rather than with every occurrence of a particular event in CICS processing (as is the case with global user exits). Task-related user exits can be used to build a resource manager interface (RMI) that enables you to access non-CICS resources, such as databases. In CICS Transaction Server, DBCTL is accessed by a CICS RMI. See also resource manager interface (RMI).

TCP. See terminal control program (TCP).

TCT. See terminal control table (TCT).

TCTSE. See terminal control table system entry (TCTSE).

TCTTE. See terminal control table terminal entry (TCTTE).

Telecommunications Access Method (TCAM). An access method used to transfer data between main storage and remote or local storage.

temporary storage (TS). A CICS facility for temporarily saving data in the form of sequential queues. A TS queue is held in main storage or on a VSAM data set on DASD. All queues not in main storage are in a single VSAM data set. A task can create a TS queue with a name selected by the task. The queue exists until deleted by a task (usually, but not necessarily, the task that created it). Compare transient data. Possible uses of temporary storage include storage of screen images for terminal paging and storage of incomplete data for suspended tasks. In general, TS queues do not require resource definition, but see temporary storage table (TST).

temporary storage table (TST). A table describing temporary storage queues and queue prefixes for which CICS is to provide recovery or security or that are located on a remote CICS system.

terminal control table (TCT). CICS control table retained to define non-VTAM terminal networks.

terminal control table system entry (TCTSE). In the TCT, an entry that is generated for each system known to the local CICS system. Using resource definition macro (RDM), the DFHTCT TYPE=SYSTEM macro defining a TCTSE must specify the applid of the remote system in the NETNAME or the SYSIDNT option. Using resource definition online (RDO), the CEDA DEFINE CONNECTION transaction defining a remote system generates a TCTSE, and must specify the applid of the remote system in the NETNAME option.

terminal control table terminal entry (TCTTE TCTE). In the TCT, an entry for each terminal known to CICS. TCTTEs are generated either during system initialization (for terminals predefined by resource definition) or when a terminal is autoinstalled. The TCTTE describes the terminal and addresses the corresponding TCTLE (RPL for VTAM terminals), the active TCA, and TIOAs; it also contains control information relating to terminal control requests issued by the CICS application program.

terminal error program (TEP). A user-replaceable CICS program used to handle error conditions that can occur when TCAM devices (in CICS Transaction Server) or BTAM terminals (in CICS/VSE) or sequential devices are used. (Node error programs must be used for VTAM-supported devices.) The terminal error program analyzes the cause of the terminal or line error that has been detected by the terminal control program. For programming information, see the Recovery and Restart Guide.

terminal input/output area (TIOA). Area that is set up by storage control and chained to the terminal control table terminal entry (TCTTE) as needed for terminal input/output operations.

terminal list table (TLT). CICS control table that allows terminal, or operator identifications, or both, to be grouped logically. See supervisory terminal functions.

terminal paging. A set of commands for retrieving pages of an oversize output message in any order.

terminal-initiated transaction routing (TTI). Transaction routing that is initiated by a request to start a remote transaction arriving from a terminal. On the basis of an installed resource definition for the transaction and possibly on decisions made in a user-written dynamic transaction routing program, the request is routed to the appropriate remote system. The transaction runs as if the terminal were attached to the transaction-owning system.

terminal-owning region (TOR). A CICS region which owns most or all of the terminals defined locally. See also application-owning region (AOR), data-owning region (DOR).

tight loop. A loop in a single program, in which the same instructions are executed repeatedly, with the result that control is never returned to CICS.

Time Sharing Option (TSO). An MVS/ESA option that provides interactive time sharing to attached terminals.

TOR. See terminal-owning region (TOR).

TP (transaction program) record (CICS Transaction Server only.) TP records are part of the user data that follows the attach FMH header in an APPC basic conversation. These records indicate the function the signon transaction program is to perform; for example, signon or signon and change password.

trace. Facility for recording CICS activity. In CICS Transaction Server, there are three destinations for trace entries: internal trace, auxiliary trace, and generalized trace facility (GTF). In CICS/VSE, there are two destinations for trace entries: internal trace and auxiliary trace.

transaction. A unit of application data processing (consisting of one or more application programs) initiated by a single request, often from a terminal. A transaction may require the initiation of one or more tasks for its execution. Contrast with task.

transaction abend code. A four-character code, defined by CICS or the user, that is used when abnormally terminating a transaction. CICS-defined transaction abend codes begin with the letter 'A'. A transaction abend code is used to indicate the cause of an error that may have occurred in CICS code or in a user program. The Messages and Codes manual contains descriptions of the transaction abend codes defined by CICS.

A transaction abend code may be placed into a transaction dump to identify it. See transaction dump code.

transaction backout. The cancellation, as a result of a transaction failure, of all updates performed by a task.

transaction deadlock. A condition in which two or more transactions cannot continue processing because each is waiting on a resource held by the other.

transaction dump. A dump of the control blocks and storage areas associated with a particular task or storage area requested by the user (see the Problem Determination Guide). A transaction dump can be requested with the EXEC CICS DUMP TRANSACTION command (see the Application Programming Reference manual). See also transaction dump code.

transaction dump code. A name of up to four characters by which a transaction dump will be known. When a transaction abend causes CICS to take a transaction dump, the associated transaction abend code is used as the transaction dump code. The Messages and Codes manual contains descriptions of the CICS transaction abend codes. A transaction dump code can be defined by CICS or the user and specifies a set of system actions held in the form of an entry in the transaction dump table (see the Problem Determination Guide). See also dump code, transaction dump, and transaction abend code.

transaction dump table (TDT). A CICS table which may contain an entry for each transaction dump code.

transaction identifier. A name of up to four characters that is specified when the transaction is defined to CICS and which is used to invoke the transaction. For example, to select a transaction, a terminal operator enters the transaction identifier.

transaction isolation. A CICS facility that offers storage protection between transactions, ensuring that a program of one transaction does not accidentally overwrite the storage of another transaction. See also storage protection. For more details, see the Recovery and Restart Guide and the Performance Guide.

transaction list table (XLT). CICS control table containing a list of transaction identifications. Depending on a system initialization specification that can be changed during system termination, the transactions in a particular XLT can be initiated from terminals during the first quiesce stage of system termination. During CICS execution the suffix of an XLT can be entered at the master terminal-the transactions in that XLT can then be enabled or disabled as a group.

transaction manager (XM). The CICS program that controls all CICS tasks.

transaction manager domain. A CICS domain that provides transaction-related services to create, terminate, purge, and inquire on tasks; and manage transaction definitions and transaction classes. The transaction manager domain is designed to provide greater reliability and improved function; it has minimal impact on users.

transaction routing. An intercommunication facility that allows terminals or logical units connected to one CICS region to initiate and to communicate with transactions in another CICS region within the same processor system or in another CICS system connected by an APPC link.

transaction security. A call to RACF each time a transaction identifier is entered at a terminal to verify that the terminal user or userid associated with that terminal is permitted to run the transaction.

transaction work area (TWA). An optional extension of the TCA, used as a work area for a given task. The TWA can be used for the accumulation of data and intermediate results during the execution of the task. When the amount of working storage for a task is relatively static, the TWA may be used if data is accessed by different programs during task processing. This approach cannot be used for multiple transactions; the TWA is released automatically at task termination.

transient data (TD). A CICS facility for temporarily saving data in the form of queues, called destinations. A TD destination is held either as a queue in a VSAM data set managed by CICS (intrapartition TD) or as a QSAM data set (in CICS Transaction Server) or a SAM data set (in CICS/VSE) outside the CICS region. See intrapartition transient data and extrapartition transient data. Contrast with temporary storage.

translator. See command language translator.

TRUE. See task-related user exit (TRUE).

TSO. See Time Sharing Option (TSO).

TWA. See transaction work area (TWA).

U

UACC. See universal access authority (UACC).

UDSA. See user dynamic storage area (UDSA).

UMT. See user-maintained data table (UMT).

unit of recovery descriptor (URD). A CICS control block that describes the progress of a unit of work through the sequence of syncpoint messages. The URD is chained off the CSA, and survives any failure of either system. It is used for recovery at CICS restart.

unit of work (UOW). A sequence of processing actions (database changes, for example) that must be completed before any of the individual actions performed by a transaction can be regarded as committed. After changes are committed (by successful completion of the UOW and recording of the syncpoint on the system log), they become durable, and are not backed out in the event of a subsequent failure of the task or system.

universal access authority (UACC). In RACF, the default access authority that applies to a resource if the user or group is not specifically permitted access to the resource. The universal access authority can be any of the access authorities.

unsolicited statistics. CICS statistics automatically gathered by CICS for a dynamically allocated and deallocated resource (for example, an autoinstalled terminal) when the resource is about to be deleted. See also interval statistics, end-of-day statistics, requested statistics, and requested reset statistics.

URM. See user-replaceable module.

use count. Number of tasks using a program concurrently. This is maintained by CICS in the program processing table.

user dynamic storage area (UDSA). A storage area in CICS Transaction Server 3.3 allocated below the 16MB line and reserved exclusively for those user application programs that execute in user-key and that reside below the 16MB line.

user exit. A point in a program at which a user exit routine may be given control. For programming information, see the Customization Guide.

user exit programming interface (XPI). A CICS interface that provides global user exit programs with access to some CICS services. XPI consists of a set of function calls that you can use in your user exit programs to extend or modify CICS system functions. For programming information, see the CICS Transaction Server Customization Guide.

user identifier (userid). A string of characters that uniquely identifies a user to a system. In CICS, a userid consists of 1-8 alphanumeric characters.

user transaction abend code (user abend code). An abend code issued by a user program or by an IBM licensed program other than CICS. See abend code.

user-key. Storage obtained by CICS in MVS open-key storage. It is for user application programs and their associated data areas. It can be accessed and modified by user applications and by CICS. See CICS-key, storage protection.

user-maintained data table (UMT). A type of CICS data table that has no CICS-supported association with its source data set after it has been loaded. Changes to the table are not automatically reflected in the source data set.

user-replaceable module (URM). Deprecated synonym for user-replaceable program.

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

V

virtual address. The address of a location in virtual storage. See virtual storage.

virtual lookaside facility (VLF). MVS/ESA facility that manages the data space associated with library lookaside (LLA).

virtual machine. The functional equivalent of a computer and its associated devices that is controlled by a user at a terminal. The IBM operating system, VM/SP, supplies a virtual machine to each logged-on user.

virtual storage. (ISO) The notional storage space that may be regarded as addressable main storage by the user of a computer system in which virtual addresses are mapped into real addresses. The size of virtual storage is limited by the addressing scheme of the computing system and by the amount of auxiliary storage available and not by the number of main storage locations.

Virtual Storage Access Method (VSAM). An access method for direct or sequential processing of fixed-and variable-length records on direct access devices.

virtual storage constraint relief (VSCR). The movement of areas of code or control blocks to storage above the 16MB line, or the reduction of code or control blocks below the 16MB line. These actions increase the storage available for user programs and data that use 24-bit addressing.

Virtual Telecommunications Access Method (VTAM). A set of programs that control communication across a network between terminals and application programs.

VLF. See virtual lookaside facility (VLF).

VSAM. See Virtual Storage Access Method (VSAM).

VSAM RLS. VSAM record-level sharing, an access mode supported by DFSMS to allow multiple applications to share data sets, with data locking at the record level. Access to data sets is through an SMSVSAM server. See also SMSVSAM.

VSAM shared resources. Buffers and strings shared by several VSAM data files. This is defined to CICS in the file control table.

VSAM work area (VSWA). An area that is acquired dynamically by the file control program when accessing a VSAM data set.

VTAM. See Virtual Telecommunications Access Method (VTAM). System initialization parameter used to include the VTAM access method. The default is YES. See the CICS Transaction Server System Definition Guide or the CICS/VSE System Definition and Operations Guide for more information.

VTAM exit trace. A CICS exit driven by VTAM to return control after servicing a request issued by CICS. Every such exit contains a trace point. This provides a way of tracing VTAM requests made from CICS.

W

warm keypoint. A keypoint written to the restart data set during controlled shutdown (after all system activity has ceased). During a subsequent warm restart, information in the warm keypoint is used to reestablish system tables to the status they had at controlled shutdown. See also keypoint.

warm start. Initialization of a CICS system using selected system status information obtained during the previous termination.

WebSphere Application Server. An e-business application deployment environment built on open standards-based technology. The Standard Edition lets you use Java servlets, JavaServer Pages and XML to transform Web sites. The Advanced Edition is a high-performance EJB server for implementing EJB components that incorporate business logic. The Enterprise Edition integrates EJB and CORBA components to build high-transaction, high-volume e-business applications.

workload management. In CICS, a method of optimizing the use of system resources by spreading workload as evenly as possible between different regions. For more details about managing workload in a CICSPlex, see the CICS Transaction Server Dynamic Transaction Routing in a CICSPlex manual.

X

XCF. See cross-systems coupling facility (XCF).

XM. See transaction manager (XM).

XPI. See user exit programming interface (XPI).

XRF. See Extended Recovery Facility (XRF). System initialization parameter that specifies that you want Extended Recovery Facility (XRF) support to be included in the system.

Index

IBM®