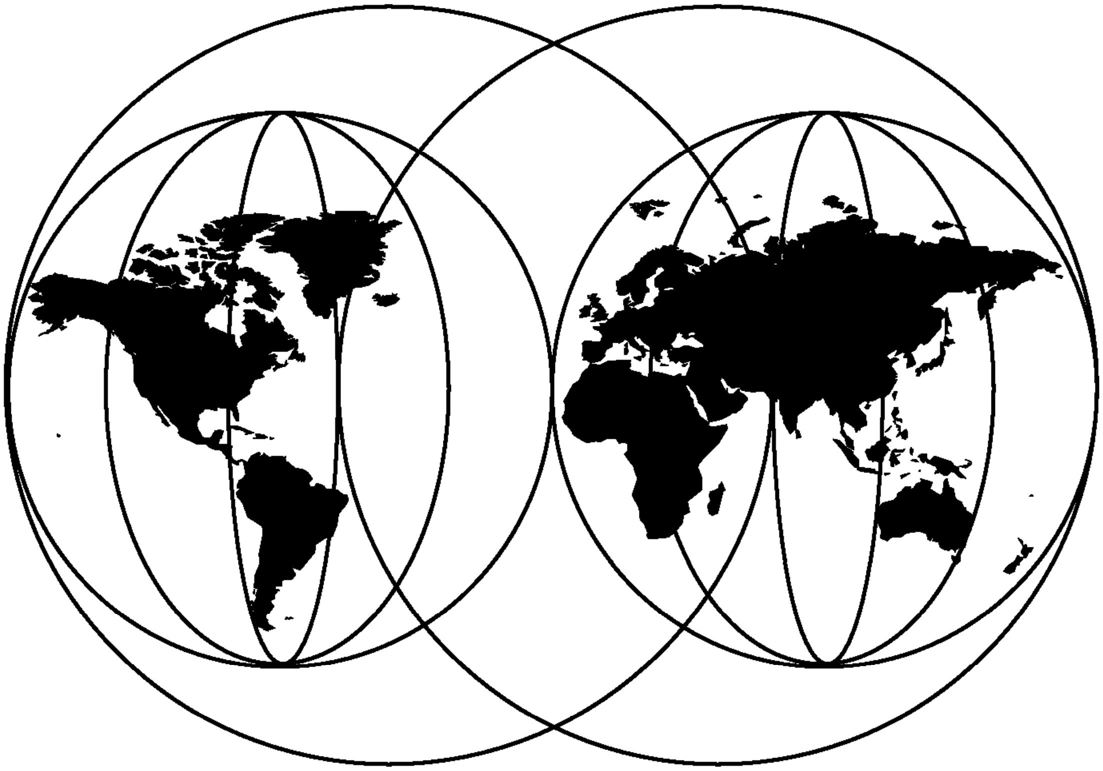
ABCs of OS/390 System Programming Volume 5

P. Rogers, G. Capobianco, D. Carey, N. Davies, L. Fadel, K. Hewitt, J. Oliveira, F. Pita, A. Salla, V. Sokal, Y. F. Tay, H. Timm



International Technical Support Organization

www.redbooks.ibm.com

SG24-5655-00

International Technical Support Organization

SG24-5655-00

ABCs of OS/390 System Programming Volume 5

April 2000

Take Note!

Before using this information and the product it supports, be sure to read the general information in Appendix E, "Special Notices" on page 383.

First Edition (April 2000)

This edition applies to OS/390 Version 2 Release 8, Program Number 5647-A01, and to all subsequent releases and modifications.

Comments may be addressed to:
IBM Corporation, International Technical Support Organization
Dept. HYJ Mail Station P099
2455 South Road
Poughkeepsie, New York 12601-5400

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

Contents

Figures

Tables

Preface

This redbook is Volume 5 of a five-volume set that is designed to introduce the structure of an OS/390 and S/390 operating environment. The set will help you install, tailor, and configure an OS/390 operating system, and is intended for system programmers who are new to an OS/390 environment.

In this Volume, Chapter 1 provides a description of a base and Parallel Sysplex. A sysplex is a collection of OS/390 systems that cooperate, using certain hardware and software products, to process work.

Chapter 2 describes the MVS System Logger. System logger is a set of services that allows an application to write, browse, and delete log data. You can use system logger services to merge data from multiple instances of an application, including merging data from different systems across a sysplex.

Chapter 3 describes Global resource serialization (GRS) which offers the control needed to ensure the integrity of resources in a multisystem environment. Combining the systems that access shared resources into a global resource serialization complex enables you to serialize resources across multiple systems.

Chapter 4 describes the operation of an MVS system which involves console operations or how operators interact with MVS to monitor or control the hardware and software and message and command processing that forms the basis of operator interaction with MVS and the basis of MVS automation.

Chapter 5 describes Automatic Restart Management (ARM) which is the key to automating the restarting of subsystems and applications (referred to collectively as applications) so they can recover work they were doing at the time of an application or system failure and release resources, such as locks, that they were holding. With an automatic restart management policy, you can optionally control the way restarts are done.

Chapter 6 describes the hardware management console (HMC) which provides a single point of control to manage your central processor complex (CPC).

Chapter 7 describes workload management which provides a way to define MVS externals and tune MVS without having to specify low-level parameters. The focus is on setting performance goals for work, and letting the Workload Manager handle processing to meet the goals.

Chapter 8 describes problem diagnosis. MVS supplies many tools and service aids that assist with problem diagnosis. These tools includes dumps and traces, while service aids includes the other facilities provided for diagnosis.

The team that wrote this redbook

This redbook was produced by a team of specialists from around the world working at the International Technical Support Organization Poughkeepsie Center.

Paul Rogers is an OS/390 specialist at the International Technical Support Organization, Poughkeepsie Center. He writes extensively and teaches IBM classes worldwide on various aspects of OS/390. Before joining the ITSO 11 years ago, he worked in the IBM Installation Support Center (ISC) in Greenford, England as OS/390 and JES support for IBM EMEA.

Guillermo Capobianco is an IT Specialist in IBM Global Services PSS Argentina. He has five years of experience working with customers on MVS, MVS-related program products, and OS/390. He is currently leading a technical group providing on-site customer support for the OS/390 platform.

David Carey is a Senior IT Availability Specialist with the IBM Support Center in Sydney, Australia, where he provides defect and nondefect support for CICS, CICSPlex/SM, MQSeries, and OS/390. David has 19 years of experience within the information technology industry, and was an MVS systems programmer for 12 years prior to joining IBM.

T. Nigel Davies is a Systems Specialist in IBM Global Services Product Support Services (PSS) in the United Kingdom. He has 10 years of IT experience in various roles, ranging from operations to PC and LAN support to mainframe systems programming. He joined IBM in 1997 with eight years of experience as a VM/VSE systems programmer, and since joining IBM has cross-trained in OS/390 systems skills. His areas of expertise include VM and VSE systems programming, installation, and technical support, and more recently, OS/390 installation and support. **Luiz Fadel**

Ken Hewitt is an IT Specialist in IBM Australia. He has over 10 years of experience working with S/390 customers in a range of roles from CE to System Engineer. His areas of expertise include I/O and OSA configuration.

Joao Natalino Oliveira

Joao Natalino de Oliveira is a certified I/T consulting specialist working for the S/390 in Brazil providing support for Brazil and Latin America. He has 24 years of experience in large systems including MVS-OS/390. His areas of expertise include performance and capacity planning, server consolidation and system programming. He has a bachelor degree in Math and Data Processing from Fundação Santo André Brazil.

Fabio Chaves Pita

Alvaro Salla has 30 years of experience in OS operating systems (since MVT). He has written several redbooks on S/390 subjects. Retired from IBM Brasil, he is now a consultant for IBM customers.

Valeria Sokal is an MVS system programmer at Banco do Brasil. She has 11 years of experience in the mainframe arena. Her areas of expertise include MVS, TSO/ISPF, SLR, and WLM.

Yoon Foh Tay is an IT Specialist with IBM Singapore PSS (S/390). He has six years of experience on the S/390 platform, providing on-site support to customers.

Hans-Juergen Timm is an Advisory Systems Engineer in IBM Global Services PSS Germany. He has 20 years of experience working with customers in the areas of MVS and OS/390, software and technical support, and planning and management. He also worked six years as an MVS Instructor in the IBM Education Centers in Mainz and Essen, Germany. His areas of expertise include implementation support for OS/390, Parallel Sysplex, UNIX System Services, and Batch Management.

Comments welcome

Your comments are important to us!

We want our redbooks to be as helpful as possible. Please send us your comments about this or other redbooks in one of the following ways:

- Fax the evaluation form found in "IBM Redbooks evaluation" on page 405 to the fax number shown on the form.
- Use the online evaluation form found at `http://www.redbooks.ibm.com/`
- Send your comments in an Internet note to `redbook@us.ibm.com`

Chapter 1. Base and Parallel Sysplex

The OS/390 sysplex has been available since 1990 when it was announced as a platform for an evolving large system computing environment. It has become the large system computing environment that offers you improved price/performance through cost effective processor technology and enhanced software. This technology builds on existing data processing skills and will run existing applications—an additional cost saver. A sysplex can also increase system availability, and at the same time, it increases your potential for doing more work.

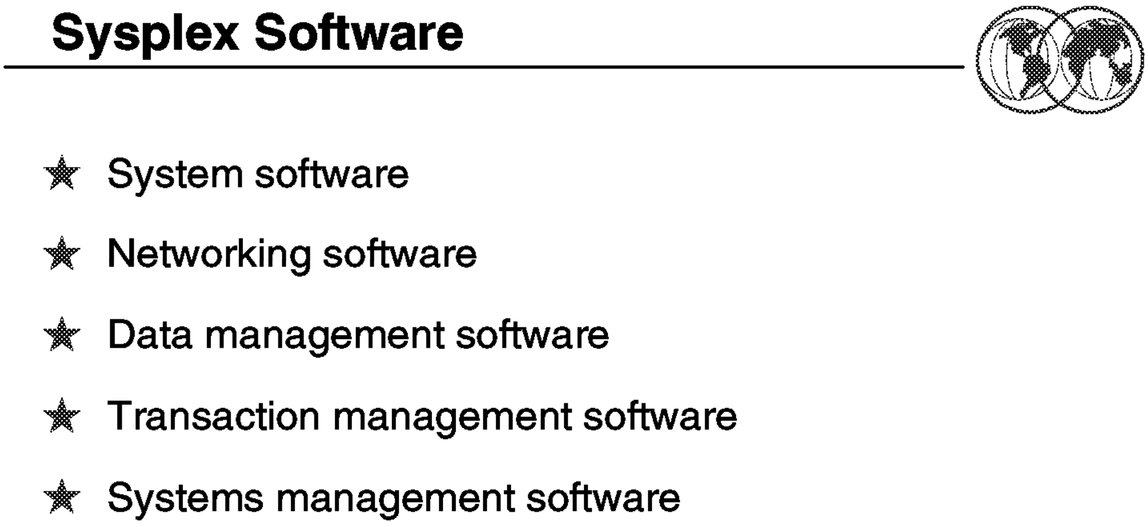
A sysplex is a collection of OS/390 systems that cooperate, using certain hardware and software products, to process work. A conventional large computer system also uses hardware and software products that cooperate to process work. A major difference between a sysplex and a conventional large computer system is the improved growth potential and level of availability in a sysplex. The sysplex increases the number of processing units and OS/390 operating systems that can cooperate, which in turn increases the amount of work that can be processed. To facilitate this cooperation, new products were created and old products were enhanced.

Since the introduction of the sysplex, IBM has developed technologies that enhance sysplex capabilities. The Parallel Sysplex architecture supports a greater number of systems and significantly improves communication and data sharing among those systems.

High performance communication and data sharing among a large number of MVS systems could be technically difficult. But with the Parallel Sysplex cluster, high performance data sharing through a new coupling technology (coupling facility) gives high performance multisystem data sharing capability to authorized applications, such as MVS subsystems. Use of the coupling facility by subsystems, such as Information Management System (IMS), ensures the integrity and consistency of data throughout the entire sysplex.

The capability of linking together many systems and providing multisystem data sharing makes the sysplex platform ideal for parallel processing, particularly for online transaction processing (OLTP) and decision support.

In short, a Parallel Sysplex cluster builds on the base sysplex capability, and allows you to increase the number of CPCs and MVS images that can directly share work. The coupling facility enables high performance, multisystem data sharing across all the systems. In addition, workloads can be dynamically balanced across systems with the help of new workload management functions.



Sysplex Software

- System software
- Networking software
- Data management software
- Transaction management software
- Systems management software

Figure 1. Sysplex software

1.1 Sysplex software

The following types of software can be part of a sysplex:

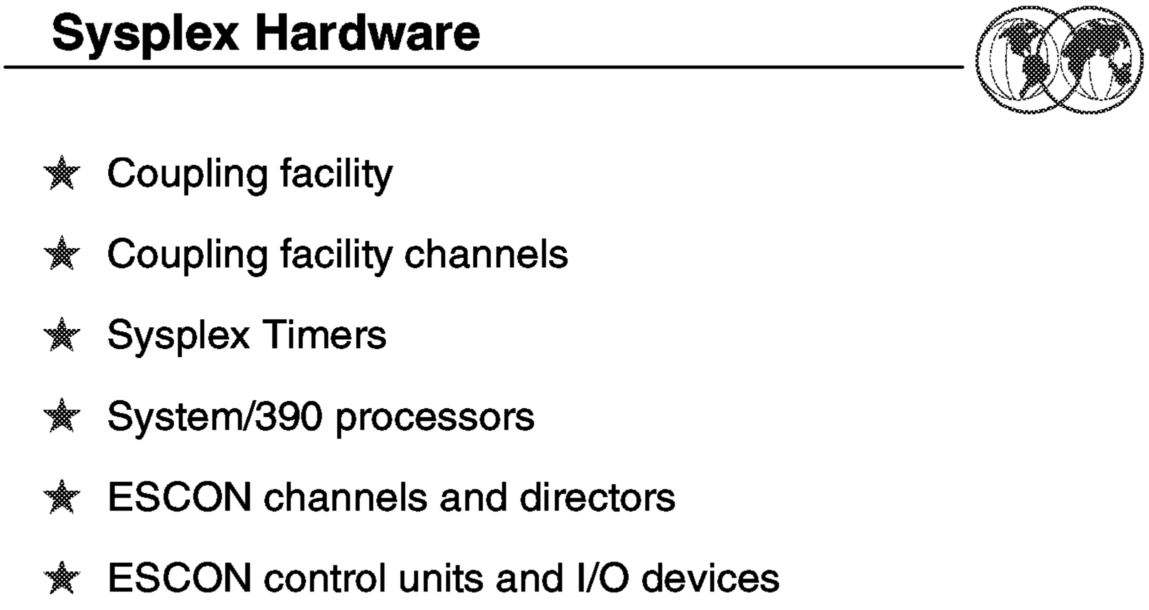
System System software is the base system software that is enhanced to support a sysplex; it includes the MVS operating system, JES2 and JES3, and DFSMS.

Networking Networking software includes Virtual Telecommunications Access Method (VTAM) which supports the attachment of a sysplex to a network.

Data management Data management software includes the data managers that support data sharing in a sysplex such as Information Management System Database Manager (IMS DB), DATABASE 2 (DB2), and Virtual Storage Access Method (VSAM).

Transaction management Transaction management software includes the transaction managers that support a sysplex such as Customer Information Control System (CICS/ESA and CICS Transaction Server) and Information Management System Transaction Manager (IMS TM).

Systems management Systems management software includes a number of software products which are enhanced to run in a sysplex and exploit its capabilities. The products manage accounting, workload, operations, performance, security, and configuration, and they make a sysplex easier to manage by providing a single point of control.



Sysplex Hardware

★ Coupling facility

★ Coupling facility channels

★ Sysplex Timers

★ System/390 processors

★ ESCON channels and directors

★ ESCON control units and I/O devices

Figure 2. Sysplex hardware

1.2 Sysplex hardware

The following types of hardware participate in a sysplex:

- Coupling facilities

  Coupling facilities enables high performance multisystem data sharing.

- Coupling facility channels

  Coupling facility channels provide high speed connectivity between the coupling facility and the central processor complexes (CPCs) that use it.

- Sysplex Timers

  A Sysplex Timer synchronizes the time-of-day (TOD) clocks in multiple CPCs in a sysplex.

- System/390 processors

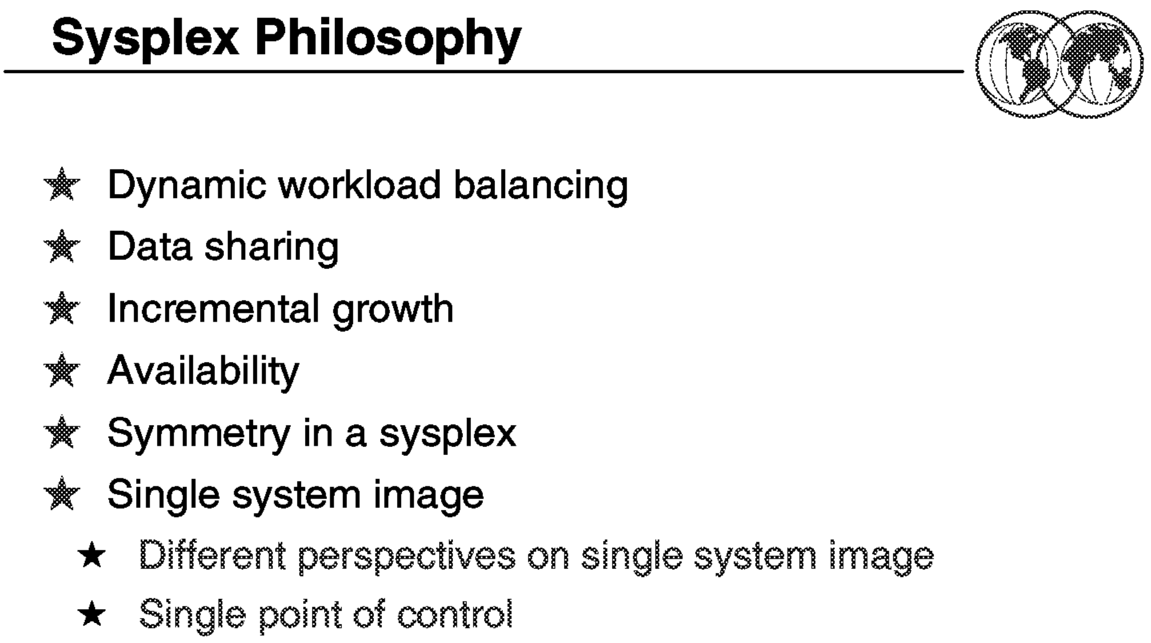
  Selected models of S/390 processors can take advantage of a sysplex. These include large water-cooled processors, air-cooled processors, and microprocessor clusters.

- ESCON channels and directors

  Enterprise Systems Connection (ESCON) channels enhance data access and communication in the sysplex. The ESCON directors add dynamic switching capability for ESCON channels.

- ESCON control units and I/O devices

ESCON control units and I/O devices in a sysplex provide the increased connectivity necessary among a greater number of systems.



Sysplex Philosophy

- ★ Dynamic workload balancing
- ★ Data sharing
- ★ Incremental growth
- ★ Availability
- ★ Symmetry in a sysplex
- ★ Single system image
  - ★ Different perspectives on single system image
  - ★ Single point of control

Figure 3. Sysplex philosophy

1.3 Sysplex philosophy

A new violinist who joins the symphony orchestra receives a copy of the score, and begins playing with the other violinists. The new violinist has received a share of the workload. Similarly in a sysplex, if you add a system, a Customer Information Control System (CICS) OLTP workload can be automatically rebalanced so that the new system gets its share, provided you have set up the correct definitions in your sysplex.

1.3.1 Dynamic workload balancing

In the CICS OLTP environment, transactions coming into the sysplex for processing can be routed to any system. CICS uses a component of OS/390 called Workload Manager (WLM), along with CICSPlex System Manager (CPSM), to dynamically route CICS transactions.

For this to happen in the sysplex, you need symmetry; the systems across which you want to automatically balance the workload must have access to the same data, and have the same applications that are necessary to run the workload.

WLM provides a way for you to have OS/390 manage your work based on performance goals. You define your performance goals in something called a service policy.

1.3.2 Data sharing

We noted earlier that in the symphony orchestra, all the instruments share the same musical score, each playing the appropriate part. You can think of the musical score as a kind of database.

Part of the definition of symmetry, as used in this redbook, is systems sharing the same resources. An important resource for systems to share is data, either in the form of a database, or data sets. Symmetry through systems sharing the same database facilitates dynamic workload balancing and availability. The coupling facility technology in the sysplex provides the data sharing capability.

You can simplify some systems management tasks, such as planning for availability, by using products like Information Management System Database Manager (IMS DB) or DATABASE 2 for OS/390 (DB2) that provide data sharing with the coupling facility technology. Systems management issues related to data sharing also include configuring and doing capacity planning for the coupling facility.

1.3.3 Incremental growth

The conductor can add violins, or other instruments, to the orchestra one by one until the desired effect is achieved. The conductor would not want to hire five more violinists if only two are needed at the moment. A sysplex exhibits the same incremental growth ability. Rather than adding capacity in large chunks, most of which might remain idle, you can add small chunks closer to the size you need at the moment.

Also, the introduction of a new violinist is nondisruptive. It′s possible (although you might see this only in the most novel of musical pieces) that the violinist could walk onto the stage in the middle of the concert, take a seat, and begin playing with the others. There is no need to stop the concert. Similarly, with a sysplex, because of symmetry and dynamic workload balancing, you can add a system to your sysplex without having to bring down the entire sysplex, and without having to manually rebalance your CICS OLTP workload to include the new system.

1.3.4 Availability

If a violinist gets sick and cannot be present for a given performance, there are enough other violinists so that the absence of one will probably not be noticeable. If a violinist decides to quit the orchestra for good, that violinist can be replaced with another. A sysplex exhibits similar availability characteristics. One of the primary goals of a sysplex is continuous availability.

You can think of availability from these perspectives: the availability of your applications, and the availability of your data.

With symmetry and dynamic workload balancing, you will find your applications can remain continuously available across changes, and your sysplex will remain resilient across failures. Adding a system, changing a system, or losing a system should have little or no impact on overall availability.

With symmetry and data sharing, using the coupling facility, you will also have enhanced database availability.

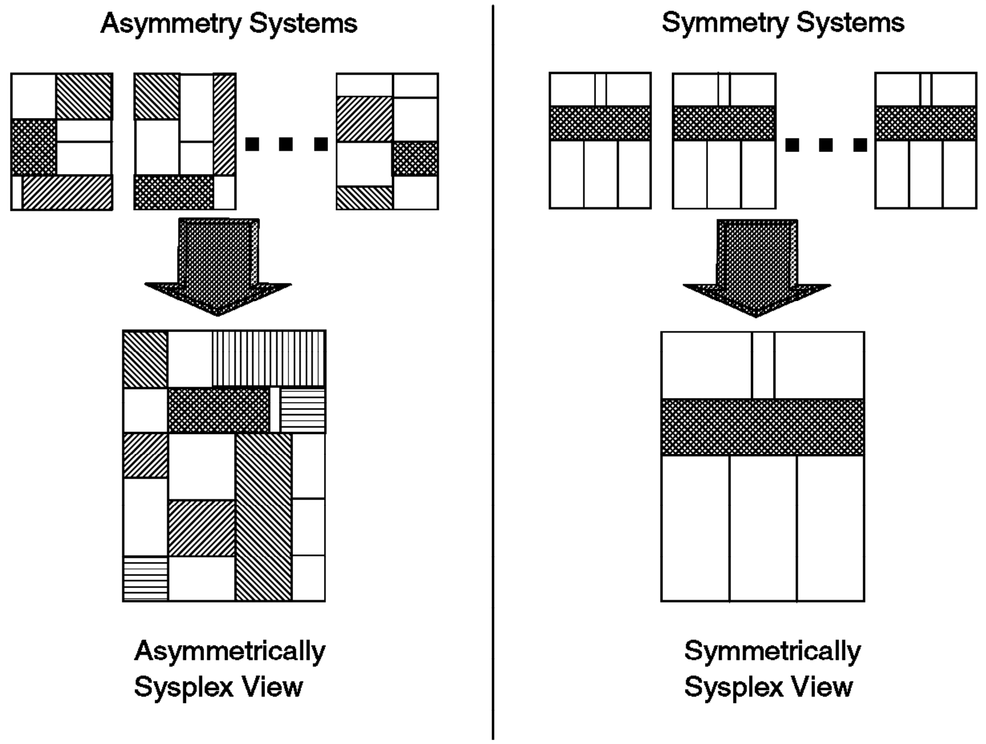
Automation plays a key role in availability. Typically, automation routines are responsible for bringing up applications, and if something goes wrong, automation handles the application′s restart. While automation doesn′t play much of a role in our symphony orchestra, the need for automation is quite important in the sysplex, for availability as well as other reasons.

A facility of OS/390 called automatic restart management provides restart capability for failed subsystems, components, and applications. Automatic restart management plays an important part in the availability of key OS/390 components and subsystems, which in turn affects the availability of data.

For example, when a subsystem such as CICS, IMS DB, or DB2 fails, it might be holding resources, such as locks, that prevent other applications from accessing the data they need. Automatic restart management quickly restarts the failed subsystem; the subsystem can then resume processing and release the resources, making data available once again to other applications.

Note that System Automation for OS/390 (SA OS/390), an IBM program product that provides automation of operator functions such as start-up, shutdown, and restart of subsystems, has awareness of OS/390 automatic restart management, so that restart actions are properly coordinated.

Asymmetry/Symmetry in a Sysplex



Asymmetry Systems

Symmetry Systems

Asymmetrically Sysplex View

Symmetrically Sysplex View

Figure 4. Asymmetry/symmetry in a sysplex

1.3.5 Symmetry, sysplex, and the symphony

You can think of a sysplex as a symphony orchestra. The orchestra consists of violins, flutes, oboes, and so on. Think of each instrument as representing a different product in the sysplex. The fact that you have several of each instrument corresponds to having several images of the same product in the sysplex.

Think of symmetry in the orchestra in the following ways:

- All the violins (or whatever other instrument) sound basically the same, and play the same musical part.
- All the instruments in the orchestra share the same musical score. Each instrument plays the appropriate part for that instrument.

Similarly in the sysplex, you can make all the systems, or a subset of them, look alike and do the same work; all the systems can access the same database, each one using the information it needs at any point in time. Symmetry provides simplicity to the sysplex. With asymmetric systems, each with its own software and hardware configurations, the complexity grows with the number of systems.

Single System Image

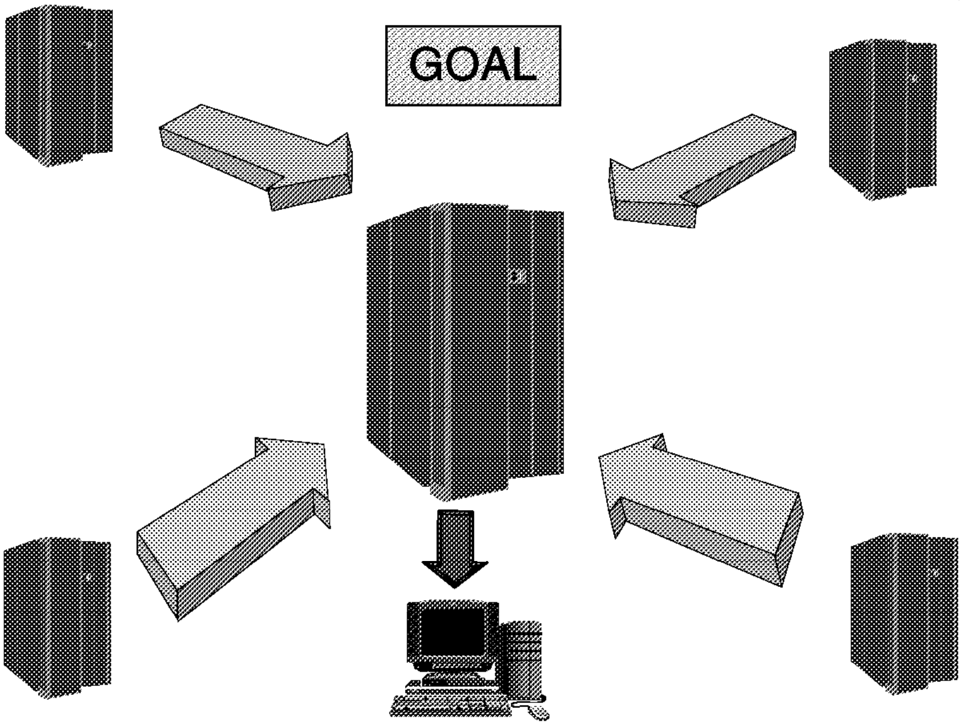


Figure 5. Single system image

1.3.6 Single system image

Think of all the violins in the symphony orchestra playing the same part. To the audience, they might sound like one giant violin. The entire orchestra is cooperating to produce the music that the audience hears. In this way, the audience perceives the orchestra as a single entity. This is a good way to picture single system image in the sysplex. You have multiple images of the same product, but they appear, and you interact with them, as one image. The entire sysplex is cooperating to process the workload. In this way, you can think of the collection of systems in the sysplex as a single entity.

Single system image is not a new concept. Many products already provide single system image capability to some degree, or have plans to implement it in the context of enterprise-wide systems management. The important point is, single system image is a key theme in a sysplex. Implementing symmetry in your sysplex facilitates single system image; symmetry facilitates your ability to manage multiple systems in the sysplex as though they were one system.

Single system image is a goal. Different IBM and non-IBM products, and even different components within products, are at different stages of development on this issue.

1.3.7 Different perspectives on single system image

The single system image goal provides different advantages depending on your perspective.

The advantage for the end user is the ability to log onto an application in the sysplex, and be able to access that application without being concerned about which system or systems the application resides on.

For example, CICS uses the VTAM generic resources function, which allows an end user to log onto one of a set of CICS terminal-owning regions (TORs), such as TOR1, TOR2, and TOR3, through a generic name, such as TOR, thus providing single system image for VTAM logons to CICS TORs. With dynamic workload management, provided by CICSPlex SM, the transaction workload is then balanced across the CICS application-owning regions (AORs), providing single system image from an application perspective.

The advantage to an operator is the ability to control the sysplex as though it were a single entity. For example, through commands with sysplex-wide scope, operators can control all the OS/390 images in the sysplex as though only one OS/390 image existed.

When TSO/E is part of a sysplex and exists on multiple sysplex members, you can assign a VTAM generic name to all TSO/E and VTAM application programs. A TSO/E and VTAM application on one OS/390 system can be known by the same generic resource as a TSO/E and VTAM application on any other OS/390 system. All application programs that share a particular generic name can be concurrently active. This means that a user can log on to a TSO/E generic name in the sysplex rather than to a particular system. The generic name can apply to all systems in the sysplex, or to a subset.

Eventually, when all the necessary products provide single system image capability, the result will be greatly improved and simplified enterprise-wide systems management. Both IBM and non-IBM products are working towards this goal.

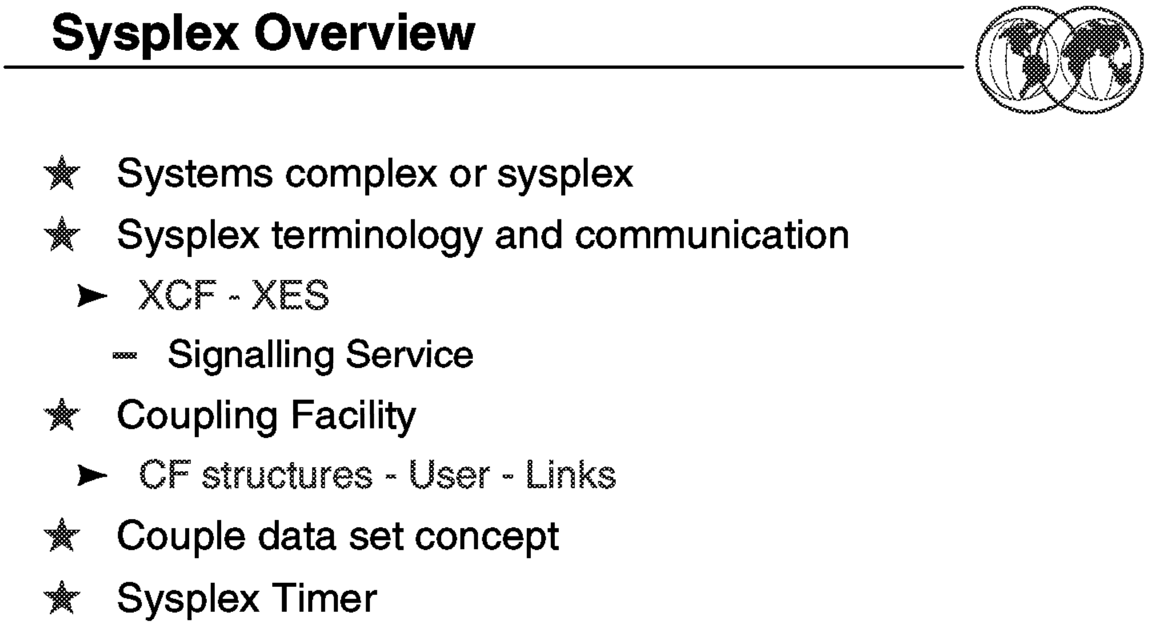
1.3.8 Single point of control

The conductor of the symphony controls the entire orchestra from the podium. The conductor does not stand by the violins and conduct them for a while, and then run over and stand by the flutes to conduct them. In a sysplex, an operator or a systems programmer should be able to control a set of tasks for the sysplex from a given workstation.

The sysplex is a little different from the symphony orchestra in that single point of control in the sysplex does not imply a single universal workstation. The object is not to control every task for every person from one place. A given individual should be able to accomplish the set of tasks pertinent to that individual′s job from one place.

Ideally, you can have multiple workstations, each tailored to a particular set of tasks; for each such workstation, you can have either a duplicate of that workstation, or some other mechanism to ensure that for every task, there is an alternative way to accomplish the task in the event the workstation, or its connection to the sysplex, fails. Even our symphony orchestra conductor has a backup; the show must go on.

IBM and non-IBM products are furthering the ability to implement single point of control through integrating operations on a workstation. IBM provides an implementation of an integrated operations workstation through Tivoli Management Environment (TME) 10.



Sysplex Overview

- ★ Systems complex or sysplex
- ★ Sysplex terminology and communication
  - ➤ XCF - XES
    - – Signalling Service
- ★ Coupling Facility
  - ➤ CF structures - User - Links
- ★ Couple data set concept
- ★ Sysplex Timer

Figure 6. Sysplex overview

1.4 Sysplex overview

Now that you′ve been introduced to the pieces that make up a sysplex, you might be wondering what a sysplex could do for you. If your data center is responsible for even one of the following types of work, you could benefit from a sysplex.

- Large business problems—ones that involve hundreds of end users, or deal with a very large volume of work that can be counted in millions of transactions per day.
- Work that consists of small work units, such as online transactions, or large work units that can be subdivided into smaller work units, such as queries.
- Concurrent applications on different systems that need to directly access and update a single database without jeopardizing data integrity and security.

A sysplex shares the processing of work across OS/390 systems, and as a result offers benefits, such as:

- Reduced cost through:
  - Cost effective processor technology
  - Continued use of large-system data processing skills without re-education
  - Protection of OS/390 application investments
  - The ability to manage a large number of systems more easily than other comparably performing multisystem environments

- Platform for continuous availability so that applications can be available 24 hours a day, 7 days a week, 365 days a year (or close to it)
- Ability to do more work
  - Greater capacity
  - Improved ability to manage response time
  - Platform for further capacity and response time advances
- Greater flexibility
  - Ability to mix levels of hardware and software
  - Ability to dynamically add systems
  - An easy path for incremental growth
  - Varied platforms for applications, including parallel, open, and client/server

Depending on your data center′s goals and needs, some of these benefits might be more attractive to you than others.

The unique characteristics of a Parallel Sysplex cluster can allow you to reduce your total cost of computing over prior offerings of comparable function and performance. Sysplex design characteristics mean that you can run your business continuously, even when it is growing or changing. You can dynamically add and change systems in a sysplex and configure them for no single points of failure. If your revenue depends on continuously available systems, a sysplex can protect your revenue. If you need to grow beyond the limits imposed by today′s technology, a sysplex lets you go beyond those limits, and helps you avoid the complex and expensive splitting and rejoining of data centers.

The innate robustness and reliability of the OS/390 operating system and System/390 processors are the foundation of a sysplex. That robustness and reliability are extended to all systems in a sysplex through cross-system workload balancing and data sharing using the coupling technologies. Therefore, applications on multiple systems can be continuously available to end users, yet the applications are shielded behind a single-system view.

The applications that run in a sysplex are the same applications you run today. Reuse of applications and data processing skills reduce the costs of application development, re-engineering, and retraining.

System Complex or Sysplex

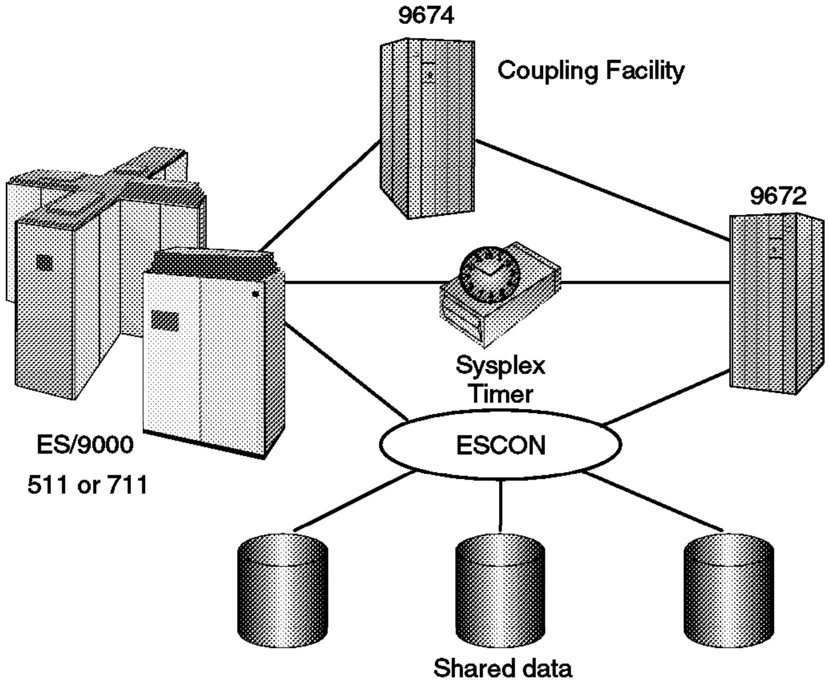
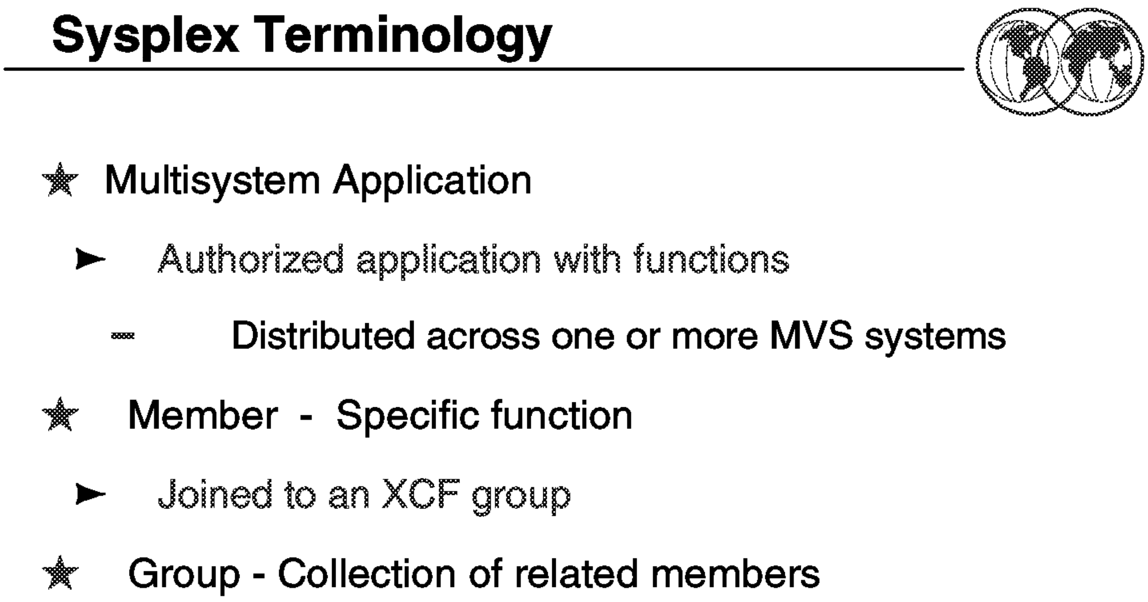


Figure 7. System complex or sysplex

1.4.1 System complex or sysplex

A **sys**tems com**plex** or sysplex is not a single product that you install in your data center. A sysplex is a collection of OS/390 systems that cooperate, using certain hardware and software products, to process workloads. The products that make up a sysplex cooperate to provide higher availability, easier systems management, and improved growth potential over a conventional computer system of comparable processing power. A conventional large computer system also uses hardware and software products that cooperate to process work. A major difference between a sysplex and a conventional large computer system is the improved growth potential and level of availability in a sysplex. The sysplex increases the number of processing units and OS/390 operating systems that can cooperate, which in turn increases the amount of work that can be processed.



Sysplex Terminology

- Multisystem Application
  - Authorized application with functions
    - Distributed across one or more MVS systems
- Member - Specific function
  - Joined to an XCF group
- Group - Collection of related members

Figure 8. Sysplex terminology

1.5 Sysplex terminology

The sysplex components are:

Application MVS services are available to authorized applications, such as subsystems and MVS components, to use sysplex services or optionally the Coupling Facility to cache data, exchange status, and access sysplex lock structures so that techniques conducive to high-performance data sharing and fast failure recovery can be implemented.

Member A member is a specific function (one or more routines) of a multisystem application that is joined to XCF and assigned to a group by the multisystem application.

Group A group is a collection of related members.

XCF Groups and Members

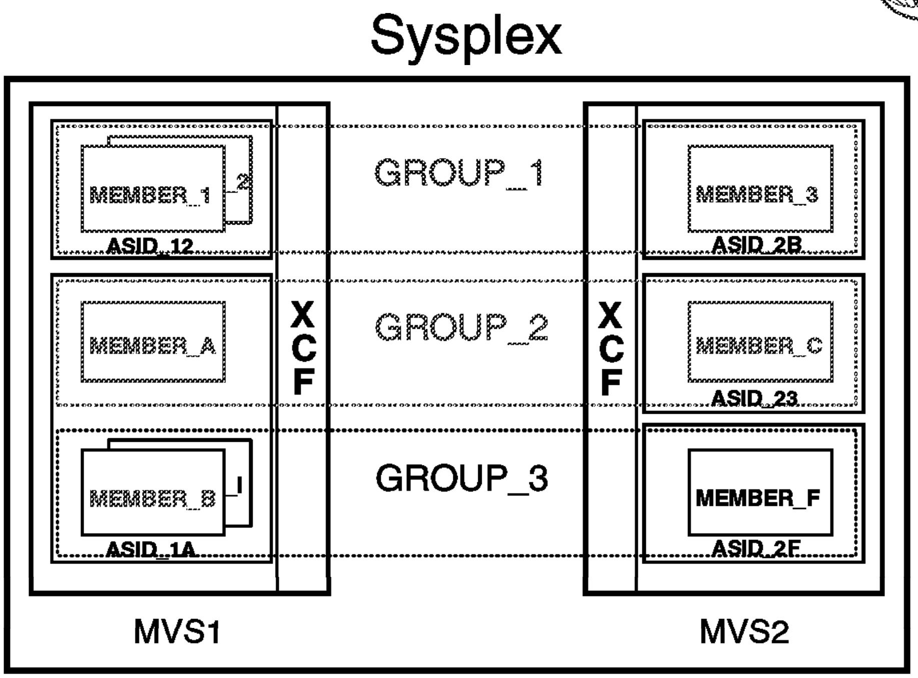


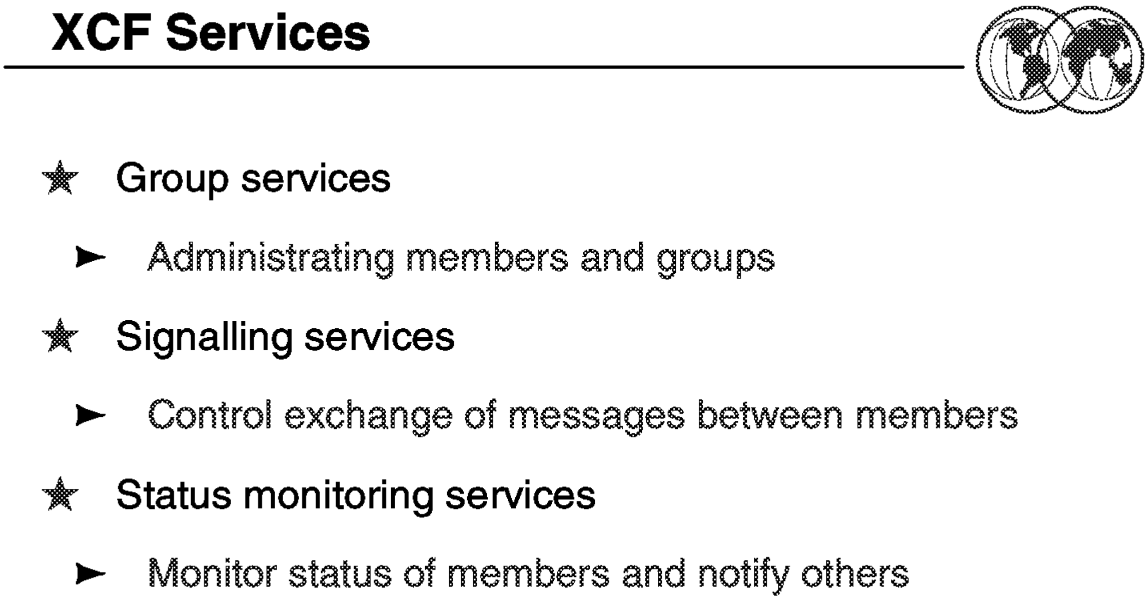
Figure 9. XCF groups and members

1.5.1 XCF groups and members

In XCF terms, a specific function (one or more routines) of a multisystem application is a *member*. A member resides on one system in the sysplex and can use XCF services to communicate with other members of the same *group*. A group in XCF is defined as the set of related members defined to XCF by the multisystem application. A group can span one or more of the systems in a sysplex and represents a complete logical entity to XCF.

Once a member becomes defined, XCF associates the member with a specific task or job step task (if specified), an address space, and a system. The association of the member with a specific task, job step task, or address space is for termination (resource clean-up) processing only. When the task, job step task, or address space that a member is associated with terminates, XCF terminates the member, thus preventing the member from further use of XCF services. Note, however, that XCF services may be requested by all tasks and SRBs in the member address space as long as the member is active.

Members of XCF groups are unique within the sysplex. However, XCF allows you to define more than one member from the same task or address space, and have those members belong to different XCF groups. This option may be used if the number of members required exceeds the maximum 511 members in a group.



XCF Services

- ★ Group services
  - ➤ Administrating members and groups
- ★ Signalling services
  - ➤ Control exchange of messages between members
- ★ Status monitoring services
  - ➤ Monitor status of members and notify others

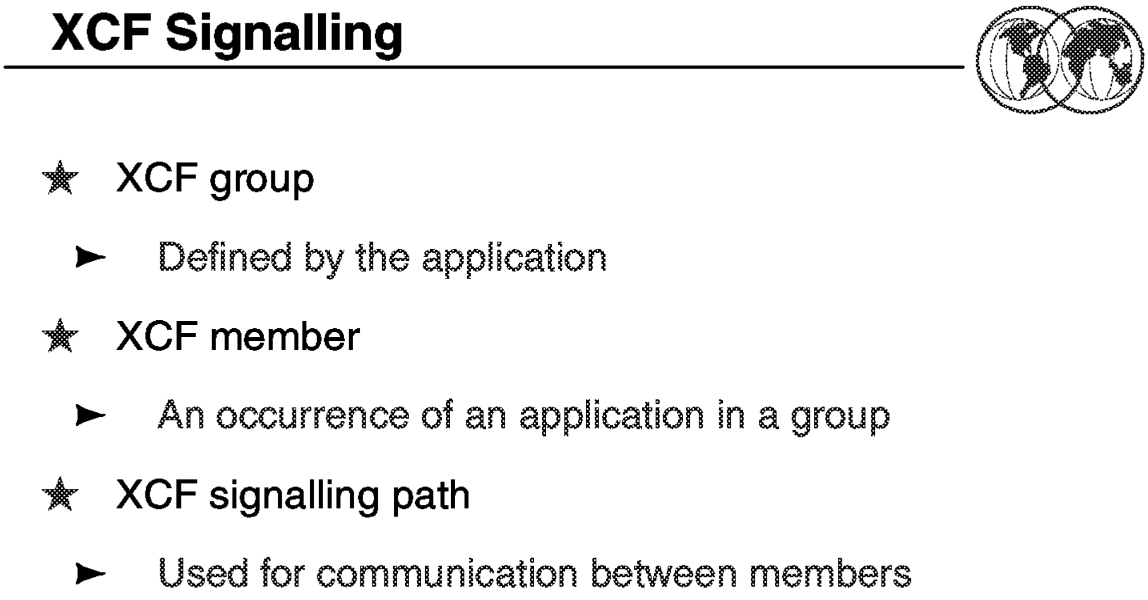
Figure 10. XCF services

1.5.2 XCF services

The MVS cross-system Coupling Facility (XCF) allows up to 32 MVS systems to communicate in a sysplex. XCF provides the MVS coupling services that allow multisystem application functions (programs) on one MVS system to communicate (send and receive data) with functions on other MVS systems.

The following services are provided by XCF:

- **Group Services** - XCF group services provide ways for defining members to XCF, establishing them as part of a group, and allowing them to find out about the other members in the group. If a member identifies a group user routine, XCF uses this routine to notify the member about changes that occur to other members of the group, or systems in the sysplex. With a group user routine, members can have the most current information about the other members in their group without having to query the system.
- **Signalling services** - Macros and an exit routine that members of a group use to communicate with other members of their group. The signalling services control the exchange of messages between members. The sender of a message requests services from XCF signalling services. XCF uses storage areas to communicate between members in the same system, and it uses the signalling paths to send messages between systems in the sysplex.
- **Status monitoring services** - Status monitoring services are authorized services that notify member exit routines of changes to the status of other members of their group and the systems in the sysplex (the group exit routine). Members can request that XCF monitor their activity (the status exit routine).



XCF Signalling

- XCF group
  - Defined by the application
- XCF member
  - An occurrence of an application in a group
- XCF signalling path
  - Used for communication between members

Figure 11. XCF signalling

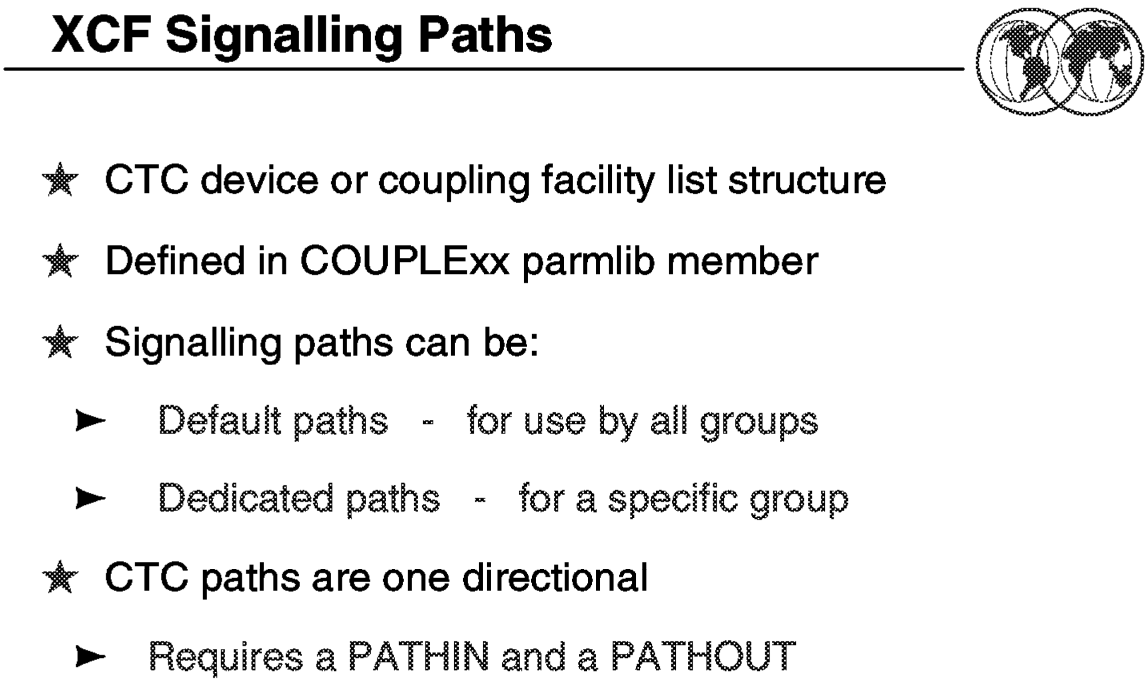
1.5.3 XCF signalling

A group in XCF is defined as the set of related members defined to XCF by the multisystem application. A group can span one or more of the systems in a sysplex and represents a complete logical entity to XCF.

Once a member becomes defined, XCF associates the member with a specific task for the group.

XCF signalling services are the primary means of communication between members of an XCF group. Members may send messages to each other as follows:

- A member sends a message by invoking the IXCMSGO macro service. Note that XCF does not necessarily deliver messages in the order in which they were sent.
- To receive the message, the receiving XCF member must be active and must have provided a message user routine. XCF gives control to the message user routine under an SRB. The message user routine receives the message by invoking the IXCMSGI macro service.



XCF Signalling Paths

- CTC device or coupling facility list structure
- Defined in COUPLExx parmlib member
- Signalling paths can be:
  - Default paths - for use by all groups
  - Dedicated paths - for a specific group
- CTC paths are one directional
  - Requires a PATHIN and a PATHOUT

Figure 12. XCF signalling paths

1.5.4 XCF signalling paths

Cross-system extended services (XES) allows subsystems running in a sysplex to have data sharing by using a coupling facility. XCF uses XES services when the XCF signalling is defined to go through a Coupling Facility list structure. JES uses XCF signalling services indirectly through JESXCF and XCF:

- Through a coupling facility, when the XCF signalling is directed to use the Coupling Facility instead of XCF CTCs
- Through XCF CTCs, when the XCF signalling is directed to use XCF CTCs instead of a Coupling Facility

XCF calls XES services when the path of communication is a Coupling Facility instead of CTCs. The communication path is determined by the PATHIN and PATHOUT definitions. The definitions for PATHIN and PATHOUT are initially defined in the COUPLExx parmlib member and can be modified by the SETXCF operator command.

OS/390 systems use XES services when the signalling paths used by XCF are a structure in a Coupling Facility.

Sysplex Communication

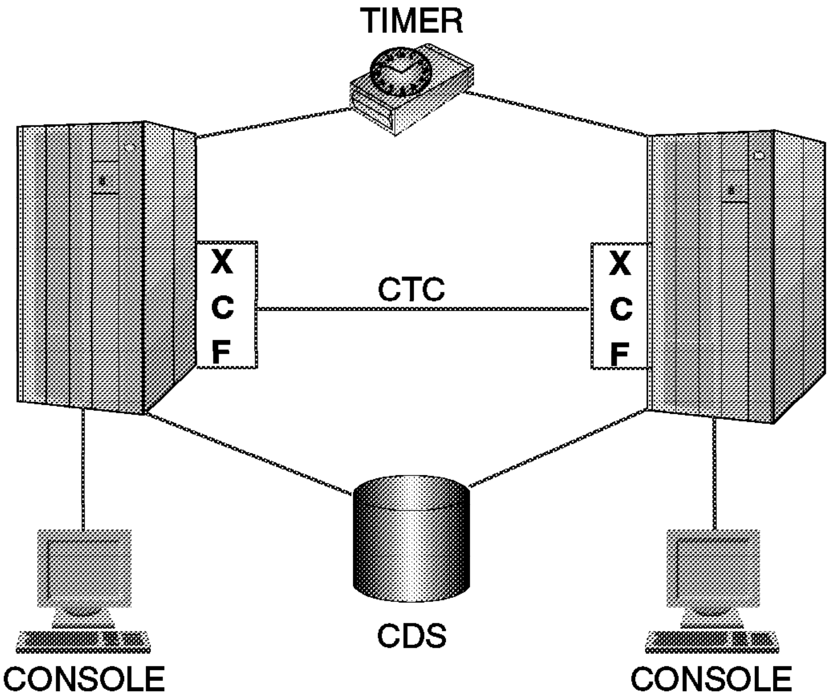


Figure 13. Sysplex communication

1.5.5 Sysplex communication

The cross-system coupling facility (XCF) component of OS/390 provides simplified multisystem management. XCF services allow authorized programs on one system to communicate with programs on the same system or on other systems. If a system fails, XCF services also provide the capability for batch jobs and started tasks to be restarted on another eligible system in the sysplex.

In a *base* sysplex, central processing complexes (CPCs) are connected through channel-to-channel (CTC) communications. In a *parallel* sysplex, central processing complexes (CPCs) are connected through channel-to-channel (CTC) and/or through a coupling facility communications.

There must be at least two operational signalling paths (one inbound and one outbound path) between each of the OS/390 systems in the sysplex.

The signalling paths can be defined through one or more CTC channels.

The other sysplex components shown are:

Timer When the sysplex consists of multiple MVS systems running on two or more processors, MVS requires that the processors be connected to the same Sysplex Timer. MVS uses the Sysplex Timer to synchronize TOD clocks across systems.

Couple data set (CDS) The sysplex couple data set is mandatory and contains XCF related data about the sysplex, systems, groups, members, and general status information. The sysplex couple data set is required to run a sysplex in either multisystem or monoplex mode.

XCF

Cross System Coupling Facility

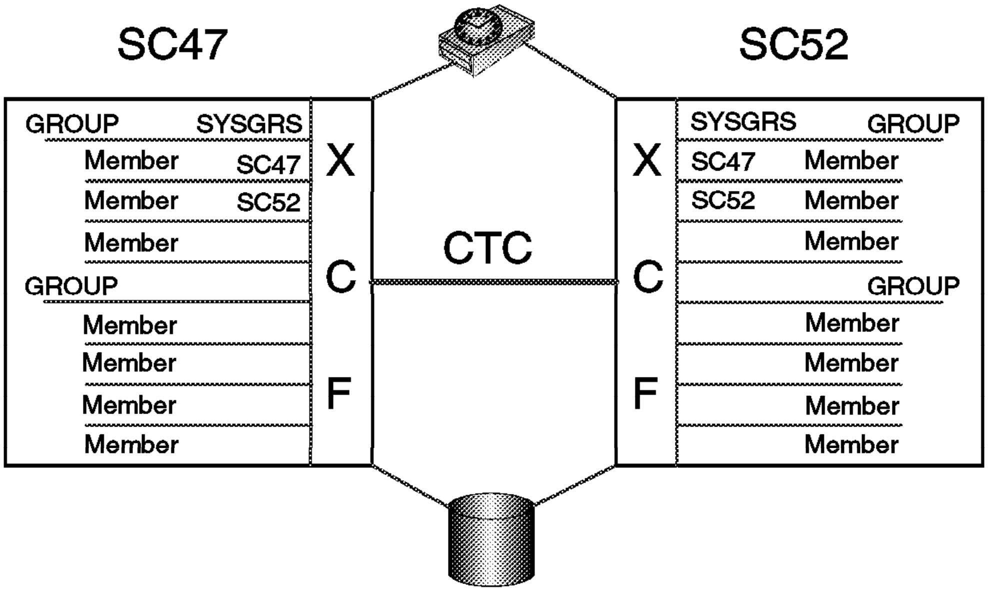


Figure 14. Cross system coupling facility

1.5.6 XCF exploiters

The following components of OS/390 are exploiters of XCF services:

- Enhanced console support

  Multisystem console support allows consolidation of consoles across multiple images. A console address space of one OS/390 image can communicate with console address spaces of all other images in the sysplex by means of XCF signalling. With this support, any console in the sysplex has the capability to view messages from any other system in the sysplex and to route commands in the same way.

  Because consoles are sysplex in scope, it may be possible to eliminate some hardware by reducing the total number of 3x74 controllers and consoles required if you manage multiple systems without sysplex.

- Global Resource Serialization (GRS)

  GRS performance is improved through its use of XCF to provide signalling services. CTCs do not have to be dedicated to GRS, and XCF will balance the I/O load across whatever number of signalling paths are available. Without a sysplex, GRS is restricted to a single primary CTC and a limited use alternate CTC. GRS also uses XCF for status monitoring, which greatly reduces operator involvement in controlling a GRS complex. All quiescing and purging of images from GRS is done automatically by XCF.

- TSO broadcast

If all images in the sysplex share the same SYS1.BRODCAST data set, the use of SYSPLXSHR(ON) will allow TSO NOTICEs to be communicated via XCF signalling, and I/O to the SYS1.BRODCAST data set is eliminated.

- JES2

  XCF is used to communicate between members of a multi-access spool (MAS). Previously, all communication between members was via the JES2 checkpoint data set. Additionally, prior to sysplex, in the event of a system failure by one member of a JES2 MAS, it was necessary for the operator to manually issue a reset command to requeue jobs that were in execution on the failing image and make them available for execution on the remaining images in the MAS complex. If running in a sysplex, this can be done automatically.

- JES3

  JES3 uses XCF services to communicate between JES3 systems in a complex. Previously, JES3 had to manage its own CTCs for this communication. This enables you to reduce the overall number of CTCs that need to be managed in your installation.

- OPC/ESA

  IBM Operations Planning and Control/ESA (OPC/ESA) is a subsystem that can automate, plan, and control the processing of a production workload. In a sysplex environment, OPC/ESA subsystems in separate OS/390 images can take advantage of XCF services for communication with each other.

- PDSE sharing

  Access to partitioned data sets extended (PDSEs) is expanded to users on any image in the sysplex. Multiple images in a sysplex can concurrently access PDSE members for input and output, but not for update-in-place. That is, any particular member of a PDSE data set, while being updated-in-place, can only be accessed by a single user. A sharer of a PDSE can read existing members and create new members or new copies of existing members concurrently with other sharers on the same image and on other images, input and output, but not for update-in-place.

- APPC/MVS

  APPC/MVS uses XCF to communicate with transaction-scheduler address spaces on the same image that APPC/MVS is running on.

- RACF sysplex communication

  RACF can be enabled for sysplex communication between members in the same sysplex. Doing this enables the subsequent commands to be propagated to members in the sysplex other than the member who issued the command, thus simplifying multisystem security management.

- RMF sysplex data server

  The RMF data server is an in-storage area which RMF uses to store SMF data, which can then be moved around the sysplex to provide a sysplex-wide view of performance, via RMF Monitor III, without having to sign on to each of the systems individually.

- Dump analysis and elimination (DAE)

  Sharing the SYS1.DAE data set between images in a sysplex can help you avoid taking multiple dumps for the same problem if it is encountered on different systems in the sysplex.

- CICS multiregion operations (MRO)

  With CICS running in a sysplex, MRO has been extended to include cross-system MRO. This is achieved using XCF services for the cross-system communication, rather than having to use VTAM ISC links, as was previously required. This can give considerable performance benefits over current ISC implementations.

- Workload Manager goal mode

  When in goal mode, WLM uses XCF for communication between WLM address spaces. In order to use WLM, you must be in a sysplex configuration, either a monoplex or a multisystem sysplex.

XCF-Signalling on Sysplex

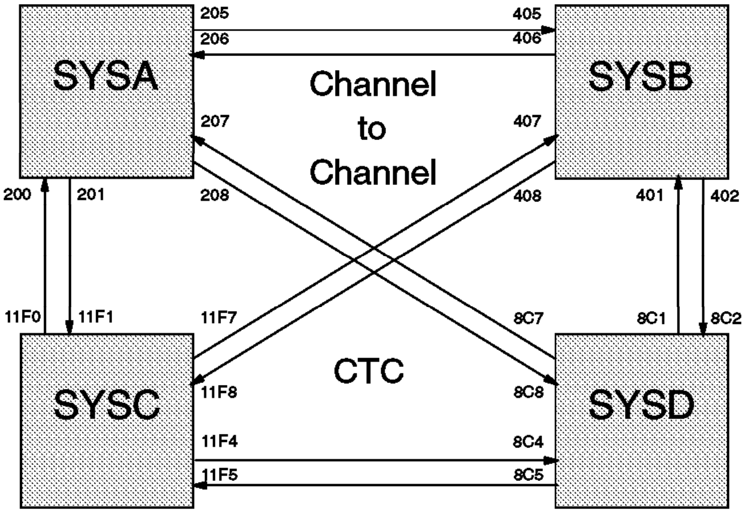


Figure 15. XCF-signalling on sysplex

1.5.7 XCF-signalling on sysplex

However signalling is achieved, the method used is transparent to exploiters of the signalling service.

Full signalling connectivity is required between all systems in a sysplex; that is, there must be an outbound and an inbound path between each pair of systems in the sysplex.

To avoid a single point of signalling connectivity failure, use redundant connections between each pair of systems, through either CTC connections or coupling facility list structures.

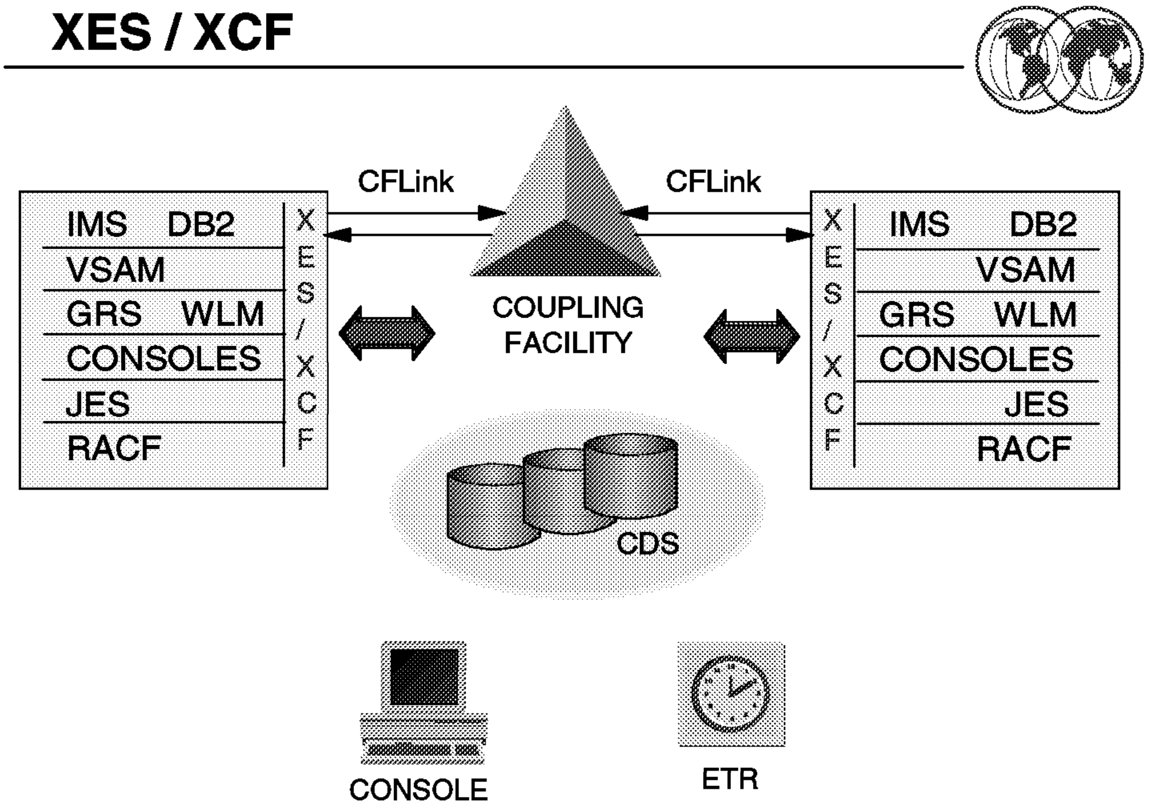


Figure 16. XES/XCF

1.5.8 XES/XCF

Cross-system extended services (XES) uses one or more coupling facilities in a Parallel Sysplex cluster to:

- Provide high-performance data sharing across the systems in a sysplex
- Maintain the integrity and consistency of shared data
- Maintain the availability of a sysplex

XES is an extension to the XCF component. XES provides the sysplex services that applications and subsystems use to share data held in the coupling facility. These services support the sharing of cached data and common queues while maintaining data integrity and consistency.

A coupling facility enables parallel processing and improved data sharing for authorized programs running in the sysplex. The cross-system extended services (XES) component of OS/390 enables applications and subsystems to take advantage of the coupling facility. XES uses one or more coupling facilities to satisfy customer requirements for:

- Data sharing across the systems in a sysplex
- Maintaining the integrity and consistency of shared data
- Maintaining the availability of a sysplex

A coupling facility is a special logical partition on CPCs in the S/390 microprocessor cluster. The coupling facility allows software on different systems in the sysplex to share data. To share data, systems must have connectivity to the coupling facility through coupling facility channels.

Systems in the sysplex that are using a coupling facility must also be able to access the coupling facility resource management (CFRM) couple data set.

Coupling Facility

CF operates as an LPAR

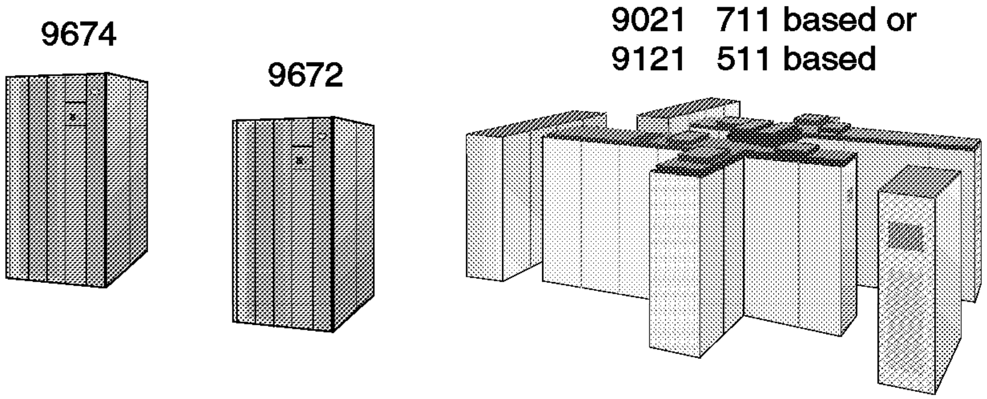


Figure 17. Coupling facility

1.6 Coupling facility

A coupling facility (CF) is a special logical partition on certain ES/9000 processors and on CPCs in the S/390 microprocessor cluster. The coupling facility allows software on different systems in the sysplex to share data. The coupling facility is the hardware element that provides high speed caching, list processing, and locking functions in a sysplex. To share data, systems must have connectivity to the coupling facility through coupling facility channels.

To enable data sharing between a CF partition and the central machines, special types of high speed CF channels are required. Two types of channel paths link a processor and a coupling facility. These coupling facility channels use high bandwidth fiber-optic links, and must be directly attached to the coupling facility.

- Coupling facility sender channels (TYPE=CFS) are the channels connecting the processor on which MVS is running to a coupling facility. These channels can be shared between partitions.
- Coupling facility receiver channels (TYPE=CFR) are the channels attached the coupling facility. These channels must be dedicated.

A CFR channel path attached to a CF partition can be connected to a CFS channel path attached to a partition in which an operating system is running. Both the coupling facility LPAR and the coupling facility channel paths must be defined to the I/O configuration data set (IOCDS).

Cross-system extended services (XES) uses one or more coupling facilities to satisfy requirements for:

- Data sharing across the systems in a sysplex

- Maintaining the integrity and consistency of shared data
- Maintaining the availability of a sysplex

Using a coupling facility in your sysplex requires both hardware and software. First, you must have a processor that supports the coupling facility control code, and secondly, a processor on which one or more MVS images will run which is capable of attaching to the coupling facility with coupling facility links. Third, you must have the the appropriate level of OS/390 that allows you to manage the coupling facility resources.

Storage in a coupling facility is divided into distinct objects called structures. Structures are used by authorized programs to implement data sharing and high-speed serialization. Structure types are cache, list, and lock, each providing a specific function to the application. Some storage in the coupling facility can also be allocated as a dedicated dump space for capturing structure information for diagnostic purposes.

You manage a coupling facility through a policy, the coupling facility resource management (CFRM) policy. CFRM allows you to specify how a coupling facility and its resources are to be used in your installation. In a CFRM policy, you supply information about each coupling facility and each coupling facility structure that you plan to use.

Your installation can define several CFRM policies, to be used at different times depending on the workload running on the sysplex. You activate a CFRM policy by issuing the operator command:

```
SETXCF START,POLICY,TYPE=CFRM,POLNAME=policy name
```

This causes the system to access the administrative policy from the CFRM couple data set, make a copy of it on the CFRM couple data set, set this as the active policy, and use the policy for managing the coupling facility resources in your sysplex.

OS/390 services allow authorized applications, such as subsystems and OS/390 components, to use the coupling facility to cache data, exchange status, and access sysplex lock structures in order to implement high performance data sharing and rapid recovery from failures. To share data, systems must have connectivity to the coupling facility through coupling facility channels. Coupling facility channels are high-bandwidth fiber-optic links that provide high speed connectivity between the coupling facility and the CPCs that use the coupling facility. Coupling facility channels are directly attached between the CPCs and the coupling facility. Systems in the sysplex that are using a coupling facility must also be able to access the coupling facility resource management (CFRM) couple data set.

A coupling facility runs as a special type of logical partition (LPAR) on certain ES/9000 and S/390 processors.

Coupling Facility Structures

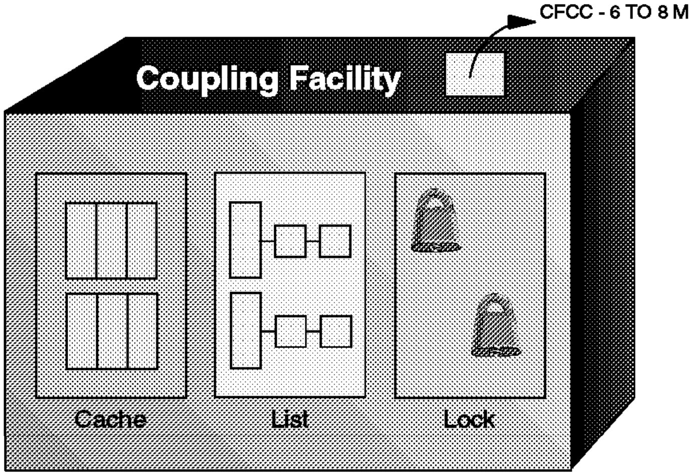


Figure 18. Coupling facility structures

1.6.1 Coupling facility structures

Within the coupling facility, storage is dynamically partitioned into structures. OS/390 services manipulate data within the structures. Each of the following structures has a unique function:

- Cache structure

  Supplies a mechanism called buffer invalidation to ensure consistency of cached data. The cache structure can also be used as a high-speed buffer for storing shared data with common read/write access.

- List structure

  Enables authorized applications to share data that is organized in a set of lists, for implementing functions such as shared work queues and shared status information.

- Lock structure

  Supplies shared and exclusive locking capability for serialization of shared resources down to a very small unit of data.

The CF architecture uses specialized hardware, licensed internal code (LIC), and enhanced OS/390 and subsystem code. The CF executes LIC called Coupling Facility Control Code (CFCC) in an LP. CFCC is loaded from the support element (SE) hard disk. The major CFCC functions are:

- Storage management
- Support for CF links
- Console services (HMC)

- Trace, logout, and recovery functions
- *Model* code that provides the list, cache, and lock structure support

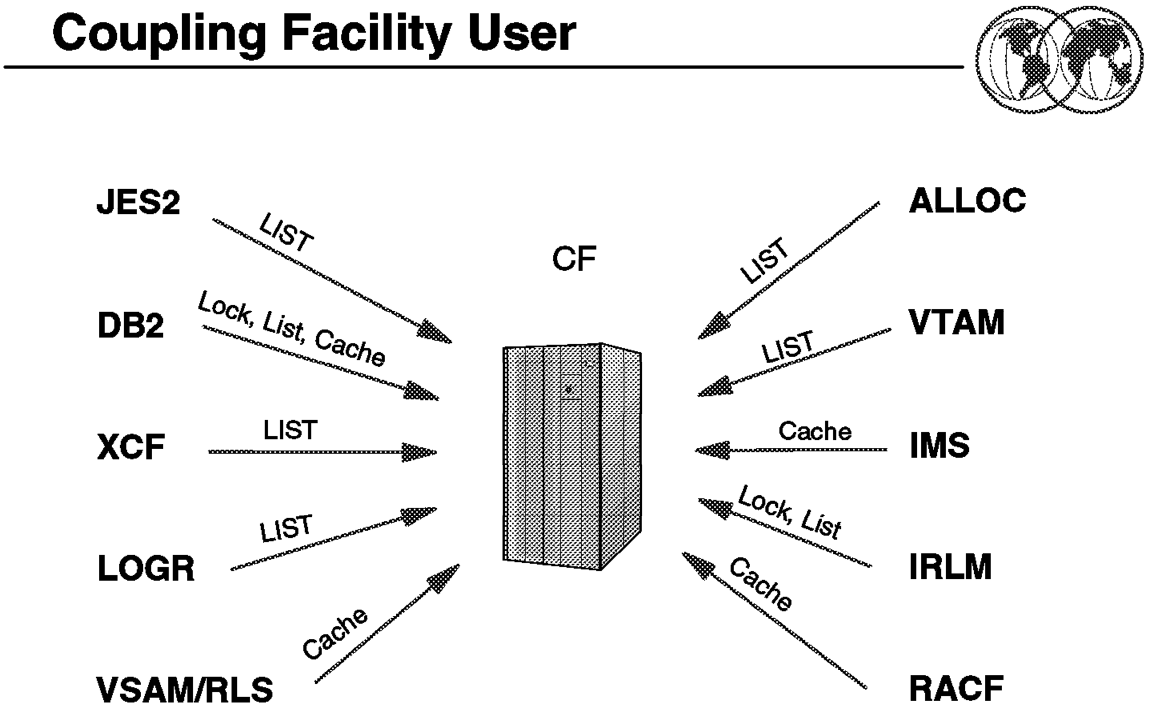


Figure 19. Coupling facility exploiters

1.6.2 Coupling facility exploiters

Authorized applications, such as subsystems and OS/390 components in the sysplex, can use the coupling facility services to cache data, share queues and status, and access sysplex lock structures in order to implement high-performance data-sharing and rapid recovery from failures. The subsystems and components transparently provide the data sharing and recovery benefits to their applications.

Some IBM data-management systems that use the coupling facility include database managers and a data access method.

Information Management System Database Manager (IMS DB) is IBM′s strategic hierarchical database manager. It is used for numerous applications that depend on its high performance, availability, and reliability. A hierarchical database has data organized in the form of a hierarchy (pyramid). Data at each level of the hierarchy is related to, and in some way dependent upon, data at the higher level of the hierarchy.

IMS database managers on different OS/390 systems can access data at the same time. By using the coupling facility in a sysplex, IMS DB can efficiently provide data sharing for more than two OS/390 systems and thereby extends the benefits of IMS DB data sharing. IMS DB uses the coupling facility to centrally keep track of when shared data is changed. IRLM is still used to manage data locking, but does not notify each IMS DB of every change. IMS DB does not need to know about changed data until it is ready to use that data.

DATABASE 2 (DB2) is IBM′s strategic relational database manager. A relational database has the data organized in tables with rows and columns.

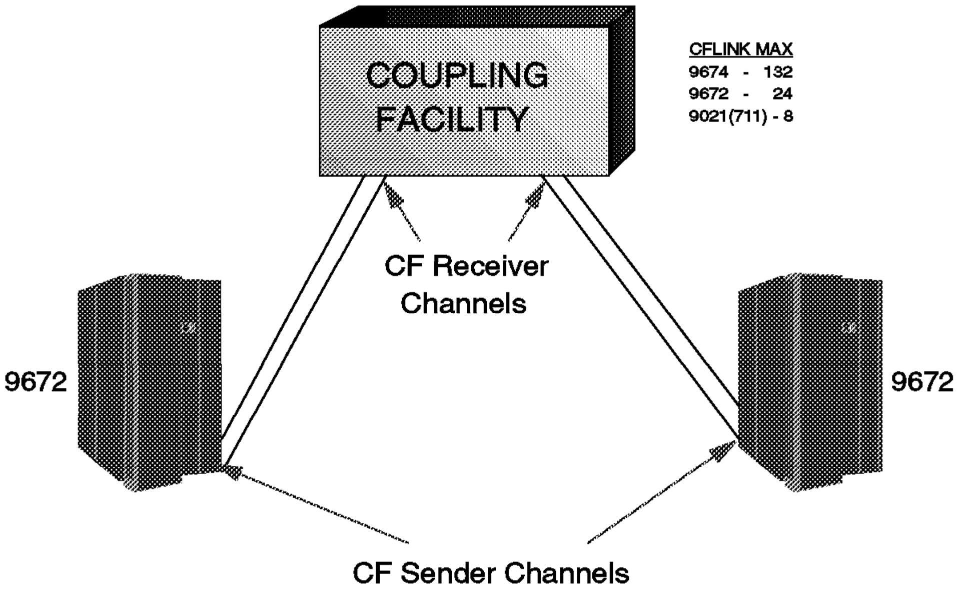
DB2 data-sharing support allows multiple DB2 subsystems within a sysplex to concurrently access and update shared databases. DB2 data sharing uses the coupling facility to efficiently lock, to ensure consistency, and to buffer shared data. Similar to IMS, DB2 serializes data access across the sysplex through locking. DB2 uses coupling facility cache structures to manage the consistency of the shared data. DB2 cache structures are also used to buffer shared data within a sysplex for improved sysplex efficiency.

Virtual Storage Access Method (VSAM), a component of DFSMS, is an access method rather than a database manager. It is an access method that gives CICS and other application programs access to data stored in VSAM-managed data sets.

DFSMS supports a new VSAM data-set accessing mode called record-level sharing (RLS). RLS uses the coupling facility to provide sysplex-wide data sharing for CICS and the other applications that use the new accessing mode. By controlling access to data at the record level, VSAM enables CICS application programs running in different CICS address spaces, called CICS regions, and in different OS/390 images, to share VSAM data with complete integrity. The coupling facility provides the high performance data-sharing capability necessary to handle the requests from multiple CICS regions.

In addition to data management systems, there are other exploiters of the coupling facility, such as Resource Access Control Facility (RACF) or the Security Server element of OS/390, and JES2. Transaction management systems also exploit the coupling facility to enhance parallelism.

Coupling Facility Links



COUPLING FACILITY

CFLINK MAX
9674 - 132
9672 - 24
9021(711) - 8

CF Receiver Channels

9672

9672

CF Sender Channels

Figure 20. Coupling facility links

1.6.3 Coupling facility links

The coupling facility is defined through Processor Resource/Systems Manager (PR/SM) panels. Once you have defined an LPAR to be a coupling facility logical partition, only the coupling facility control code can run in that partition. When you activate the coupling facility LPAR, the system automatically loads the coupling facility control code from the processor controller or the support element of the processor.

Two types of channel paths link a processor and a coupling facility:

- Coupling facility sender channels (TYPE=CFS) are the channels connecting the processor on which OS/390 is running to a coupling facility. These channels can support PR/SM ESCON Multiple Image Facility (EMIF), and therefore can be shared between partitions.
- Coupling facility receiver channels (TYPE=CFR) are the channels attached to the coupling facility. These channels must be dedicated.

Both the coupling facility LPAR and the coupling facility channel paths must be defined to the I/O configuration data set (IOCDS). Hardware configuration definition (HCD) provides the interface to accomplish these definitions and also automatically supplies the required channel control unit and I/O device definitions for the coupling facility channels.

The level of the coupling facility control code (CFLEVEL) that is loaded into the coupling facility LPAR determines what functions are available for the exploiting applications. Different levels provide new functions and enhancements that an application might require for its operation. Different levels also provide model-dependent limits that might place limitations on how the coupling facility can be used.

Couple Data Set Concept

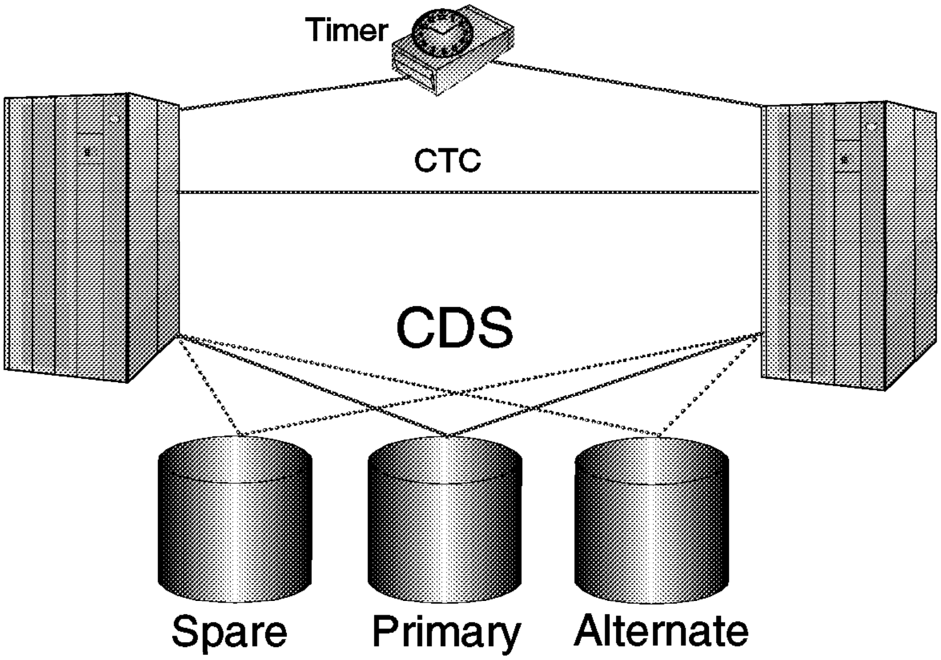


Figure 21. Couple data set concept

1.7 Couple data set concept

A sysplex requires a direct access storage device (DASD) data set (couple data set) to be shared by all systems to store status information about the OS/390 images, XCF groups, and the XCF group members running in the sysplex, as well as general status information. This is called the sysplex, or the XCF, couple data set. A sysplex couple data set is always required for a multisystem sysplex and for most single-system sysplexes. However, you can define a single system sysplex that does not require a sysplex couple data set.

When the Workload Manager and sysplex failure management features are enabled to help manage resources and workload for the sysplex, you will need to define additional couple data sets to store policy-related information for use by these components.

A policy is a set of rules and actions that systems in a sysplex are to follow when using certain OS/390 services. A policy allows OS/390 to manage specific resources in compliance with your system and resource requirements, but with little operator intervention. A policy can be set up to govern all systems in the sysplex or only selected systems. You might need to define more than one policy to allow for varying workloads, configurations, or other installation requirements at different times. For example, you might need to define one policy for your prime shift operations and another policy for other times. Although you can define more than one policy of each type (except for system logger) only one policy of each type can be active at a time. For system logger, there is only one LOGR policy in the sysplex.

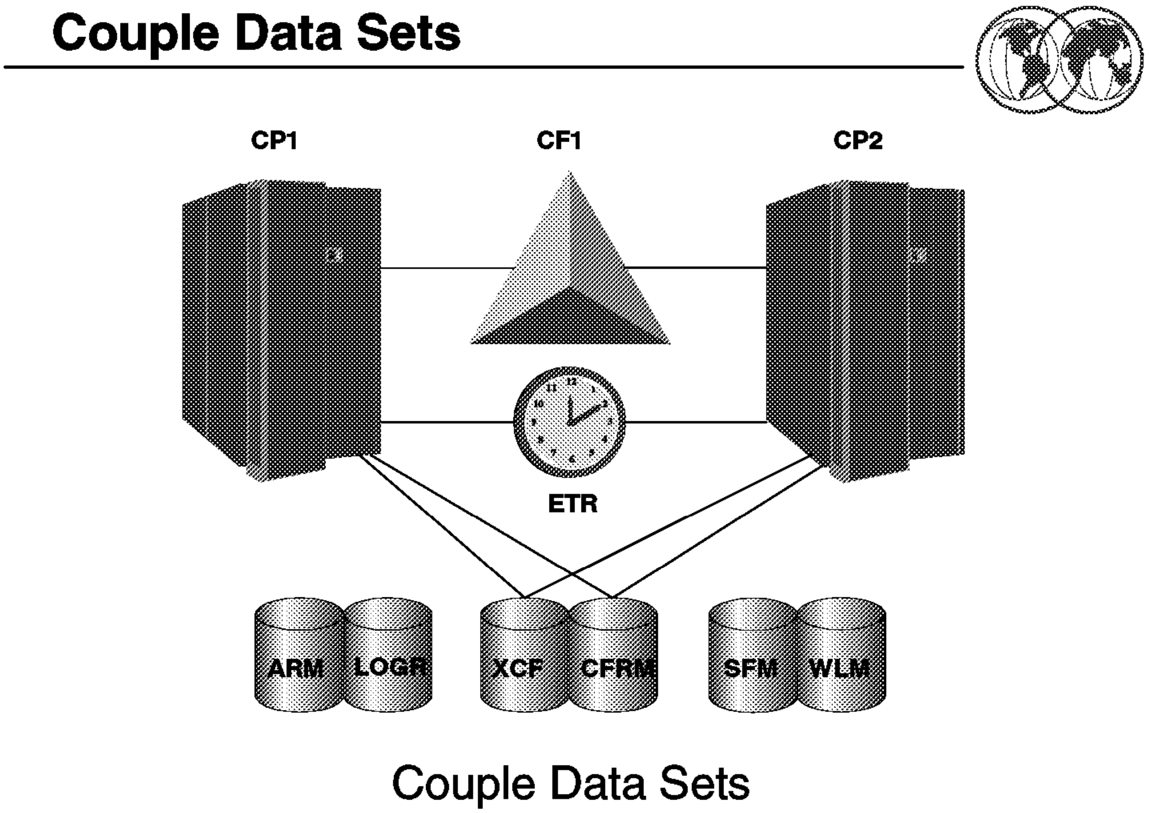


Figure 22. Couple data sets

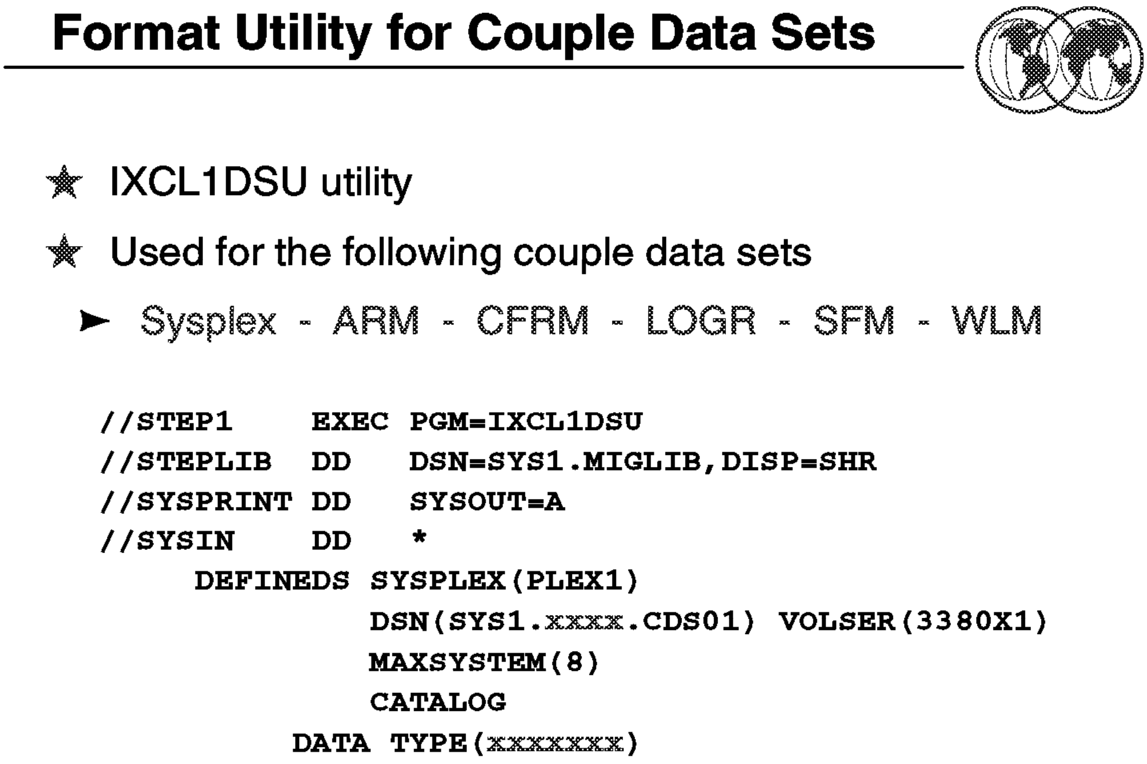
1.7.1 Couple data sets

To manage certain aspects of your sysplex using policies, you can install one or more additional data sets, for:

- Coupling facility resource management (CFRM) policy, to define how OS/390 is to manage coupling facility resources
- Sysplex failure management (SFM) policy, to define how OS/390 is to manage system and signalling connectivity failures and PR/SM reconfiguration actions
- Workload management (WLM) policy, to define service goals for workloads
- Automatic restart management (ARM) policy, to define how to process restarts for started tasks and batch jobs that have registered with automatic restart management
- System logger (LOGR) policy, to define, update, or delete structure or log stream definitions

Each of these policies, along with real-time information about the sysplex and the resources being managed when the policy is in effect, resides in a couple data set. Policy specifications cannot reside in the sysplex couple data set, but you can combine data for different policies in the same couple data set.

Before you can define and activate a policy, you must format a couple data set to hold the policy and ensure that the data set is available to the systems in the sysplex that need it. All couple data sets can be formatted using the couple data set format utility, IXCL1DSU, which resides in SYS1.MIGLIB.



Format Utility for Couple Data Sets

- IXCL1DSU utility
- Used for the following couple data sets
  - Sysplex - ARM - CFRM - LOGR - SFM - WLM

```
//STEP1     EXEC PGM=IXCL1DSU
//STEPLIB   DD   DSN=SYS1.MIGLIB,DISP=SHR
//SYSPRINT DD   SYSOUT=A
//SYSIN     DD   *
     DEFINEDS SYSPLEX(PLEX1)
              DSN(SYS1.xxxx.CDS01) VOLSER(3380X1)
              MAXSYSTEM(8)
              CATALOG
          DATA TYPE(xxxxxxx)
```

Figure 23. Format utility for couple data sets

1.7.2 Format utility for couple data sets

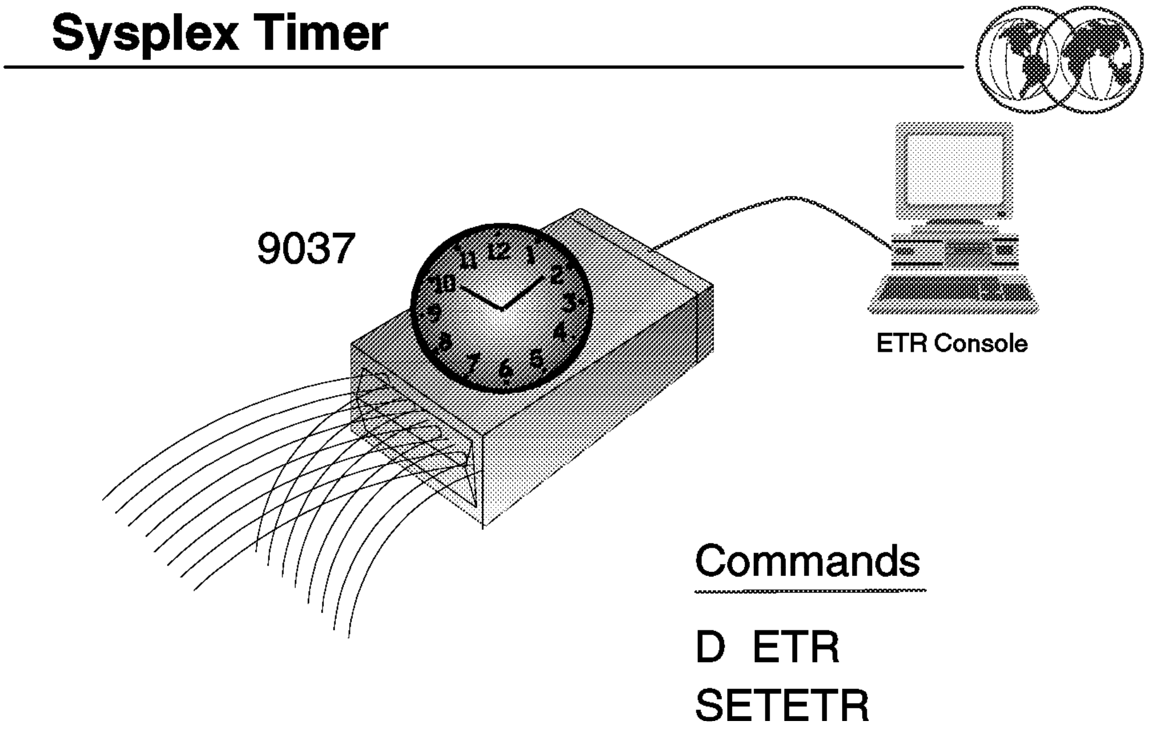
IBM provides the couple data set format utility to allocate and format the DASD couple data sets for your sysplex.

The name of the XCF couple data set format utility is IXCL1DSU. This program resides in SYS1.MIGLIB (which is logically appended to the LINKLIST), which makes it available through STEPLIB on a pre-OS/390 system.

IXCL1DSU in OS/390 allows you to format all types of couple data sets for your sysplex. The utility contains two levels of format control statements. The primary format control statement, DEFINEDS, identifies the couple data set being formatted. The secondary format control statement, DATA TYPE, identifies the type of data to be supported in the couple data set—sysplex data, automatic restart management (ARM) data, CFRM data, SFM data, WLM data, or LOGR data. In particular, note the recommendations about not over-specifying the size or the parameter value in a policy, which could cause degraded performance in a sysplex.

The control statements for the utility programs follow standard conventions for JCL statements. The utility program accepts 80-byte records that can contain one or more parameters per record.

Use the XCF couple data set format utility to create and format all sysplex couple data sets prior to IPLing a system that is to use the sysplex couple data sets. Other types of couple data sets do not have to be formatted prior to IPLing the system.



Sysplex Timer

Commands

D ETR
SETETR

Figure 24. Sysplex Timer

1.8 Sysplex Timer

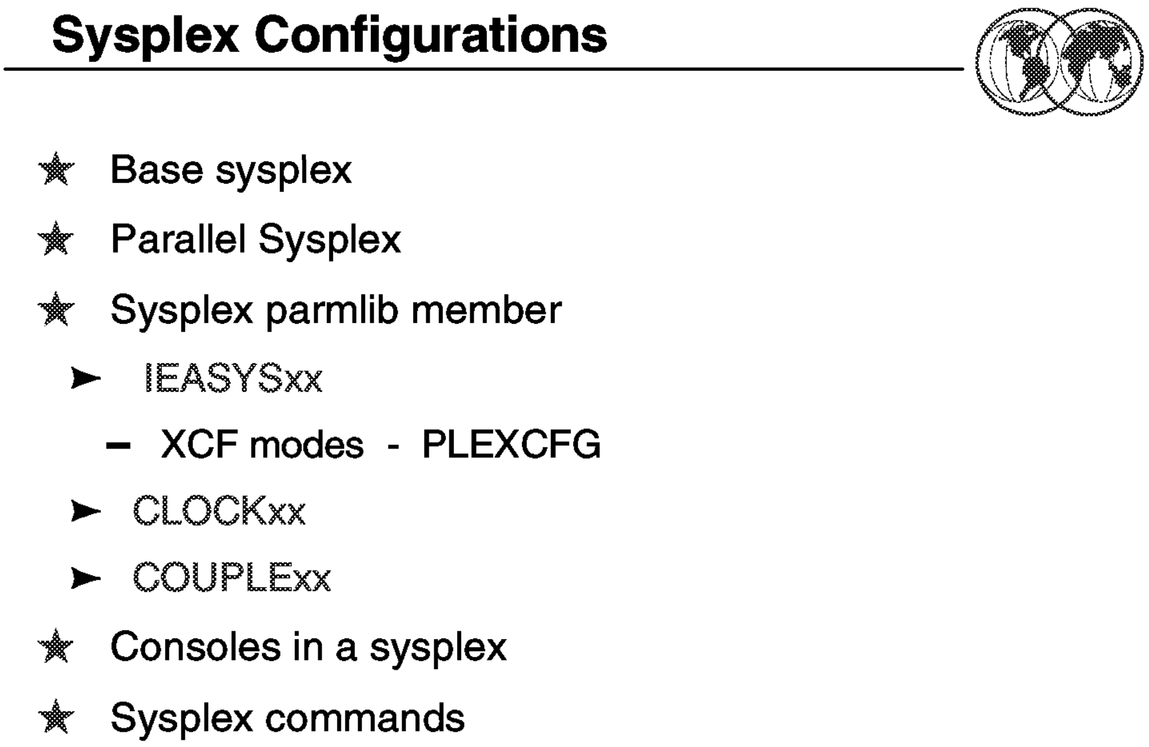
When the sysplex consists of multiple OS/390 systems running on two or more processors, OS/390 requires that the processors be connected to the same Sysplex Timer. OS/390 uses the Sysplex Timer to synchronize time-of-day (TOD) clocks across systems in a sysplex that run on different processors. The time stamp from the Sysplex Timer is a way to monitor and sequence events within the sysplex.

Use timing services to determine whether the basic or extended time-of-day (TOD) clock is synchronized with an External Time Reference hardware facility (ETR). ETR is the MVS generic name for the IBM Sysplex Timer. Timing services is also used to obtain the present date and time, convert date and time information to various formats, or for interval timing. Interval timing lets you set a time interval, test how much time is left in the interval, or cancel the interval. Use communication services to send messages to the system operator, to TSO/E terminals, and to the system log.

Several processors can share work in a data processing complex. Each of these processors has access to a TOD clock. Thus, when work is shared among different processors, multiple TOD clocks can be involved. However, these clocks might not be synchronized with one another. The External Time Reference (ETR) is a single external time source that can synchronize the TOD clocks of all processors in a complex.

For programs that are dependent upon synchronized TOD clocks in a multisystem environment, it is important that the clocks are in ETR synchronization. Use the STCKSYNC macro to obtain the TOD clock contents and determine if the clock is synchronized with an ETR. STCKSYNC also provides an optional parameter, ETRID, that returns the ID of the ETR source with which the TOD clock is currently synchronized.

For a multisystem sysplex defined on a single processor (under PR/SM or VM) the SIMETRID parameter in the CLOCKxx parmlib member must specify the simulated Sysplex Timer identifier to synchronize timings for the OS/390 systems.



Sysplex Configurations

- Base sysplex
- Parallel Sysplex
- Sysplex parmlib member
  - IEASYSxx
    - XCF modes - PLEXCFG
  - CLOCKxx
  - COUPLExx
- Consoles in a sysplex
- Sysplex commands

Figure 25. Sysplex configurations

1.9 Sysplex configurations

The remainder of this chapter describes the various sysplex configurations, which components exploit XCF services, and how system consoles are used to enter sysplex related commands.

Base Sysplex

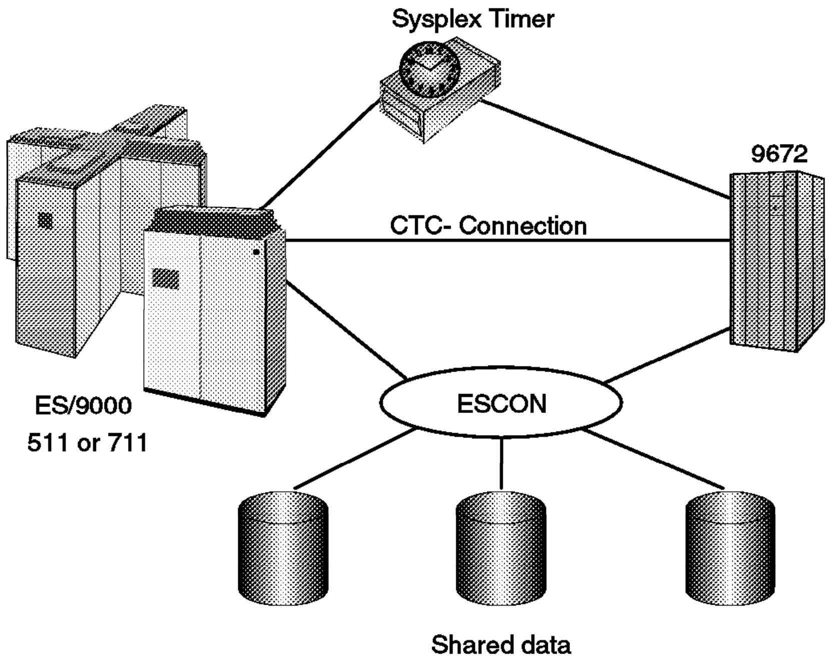


Figure 26. Base sysplex

1.9.1 Base sysplex

The base sysplex supports multisystem management through the cross-system Coupling Facility (XCF) component of OS/390. XCF services allow authorized applications on one system to communicate with applications on the same system or on other systems.

In a base sysplex, CPCs connect by channel-to-channel communications and a shared couple data set to support the communication. Full signalling connectivity is required between all images in the sysplex. That is, there must be at least one inbound path (PATHIN) and one outbound path (PATHOUT) between each pair of images in the sysplex. In a base sysplex, XCF uses CTCs for signalling. XCF CTCs can be dynamically allocated and deallocated using the OS/390 system command `SETXCF START,PATHIN/PATHOUT`. To avoid unplanned system outages due to signalling path failures, define multiple CTCs to XCF. When more than one CPC is involved, a Sysplex Timer synchronizes the time on all systems.

The base sysplex is similar to a loosely coupled configuration in that more than one CPC (possibly a tightly coupled multiprocessor) shares DASD and is managed by more than one OS/390 image. A sysplex is different from a loosely coupled configuration because through XCF, there is a standard communication mechanism for OS/390 system applications.

Next we will discuss some of the features you can take advantage of in a base sysplex environment that uses XCF communication services.

Parallel Sysplex

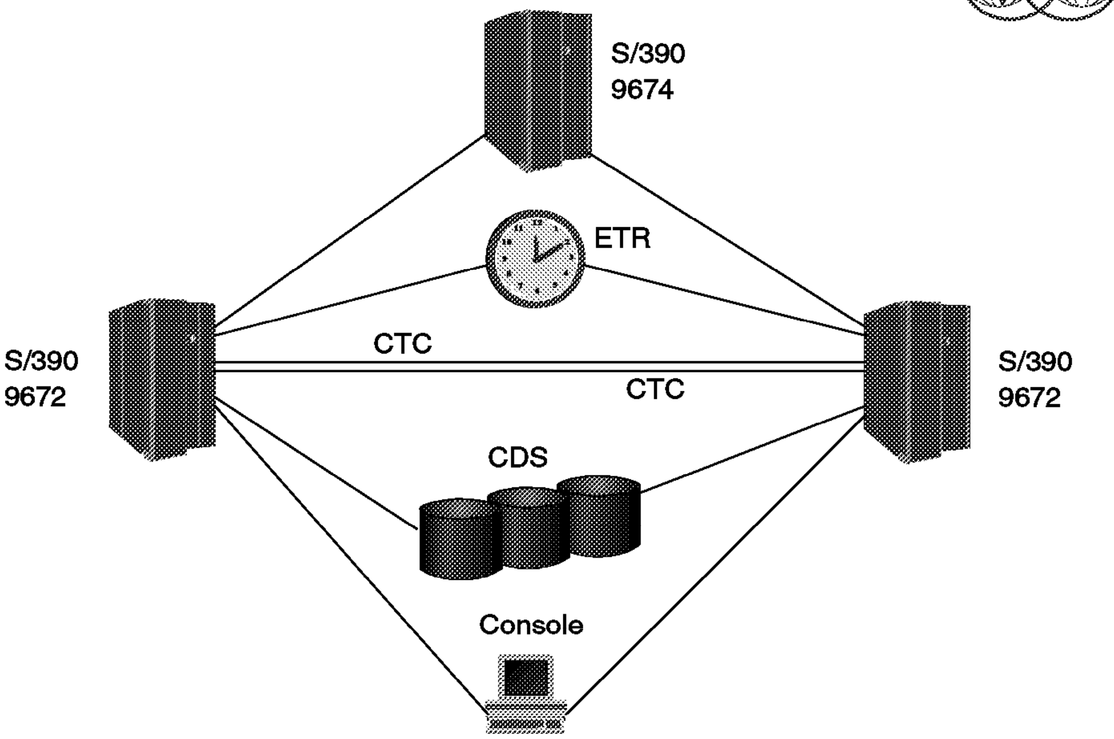


Figure 27. Parallel Sysplex

1.9.2 Parallel Sysplex

The members of a Parallel Sysplex cluster use XCF signalling to communicate with each other. In a base sysplex, this is done through CTC connections. With Parallel Sysplex, signalling can be through Coupling Facility structures, CTCs, or a combination of CTCs and signalling structures.

Implementing signalling through Coupling Facility list structures provides significant advantages in the areas of systems management and recovery and, thus, provides enhanced availability for sysplex systems.

While a signalling path defined through CTC connections must be exclusively defined as either outbound or inbound, you can define a Coupling Facility list structure so that it is used for both outbound and inbound signalling paths, and OS/390 automatically establishes the paths. For example, if you define a list structure for outbound message traffic, OS/390 automatically establishes a signalling path with every other system that has the structure defined for inbound message traffic. Similarly, if you define a list structure for inbound traffic, OS/390 automatically establishes a signalling path with every other system that has the structure defined for outbound traffic.

Because a single list structure can be defined for both inbound and outbound traffic, you can use the same COUPLExx parmlib member for each system in the sysplex. A CTC connection, in contrast, cannot be defined to a single system as both inbound and outbound. Therefore, with CTCs, you must specify a unique COUPLExx member for each system in the sysplex or configure your systems so that they can use the same COUPLExx member by over-specifying the number of devices for signalling paths and tolerating failure messages related to unavailable devices and other configuration errors.

Implementing signalling through Coupling Facility list structures also enhances the recovery capability of signalling paths and reduces the amount of operator intervention required to run the sysplex.

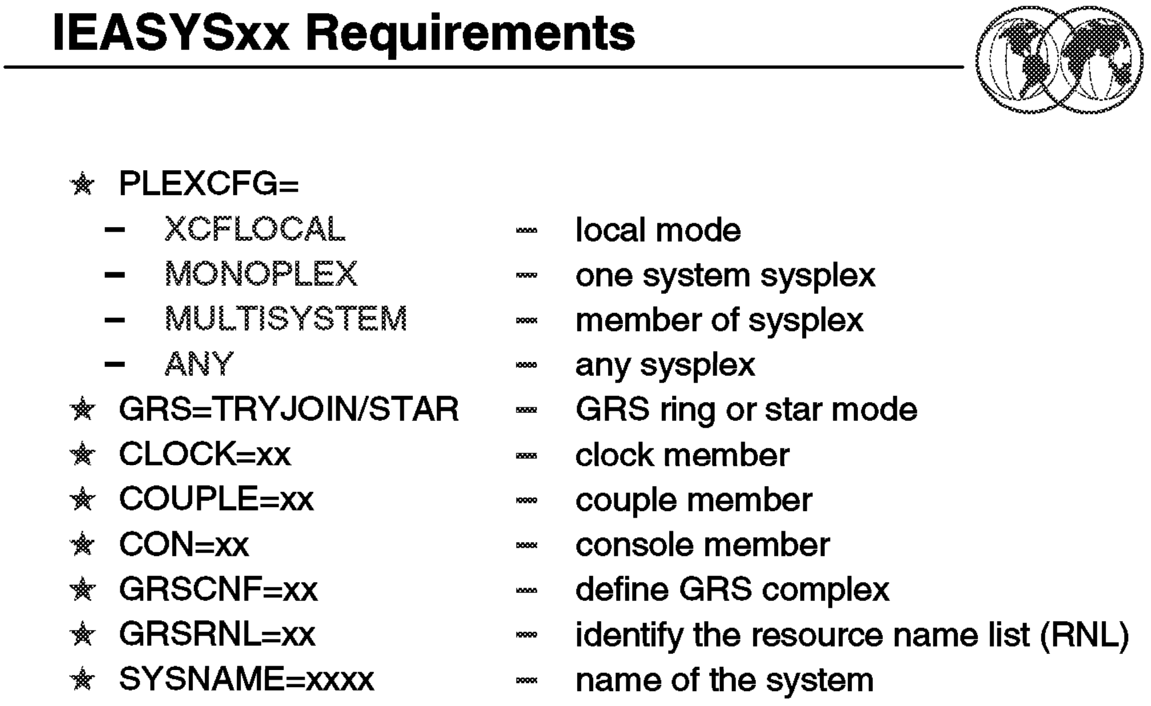
If signalling is implemented through Coupling Facility list structures, and if a Coupling Facility that holds a list structure fails or if connectivity to a Coupling Facility that holds a list structure is lost, OS/390 can rebuild the list structure in another available Coupling Facility and reestablish signalling paths.

If the list structure itself fails, OS/390 can rebuild it in the same Coupling Facility or in another available Coupling Facility and then reestablish signalling paths.

The Parallel Sysplex supports a greater number of systems and significantly improves communication and data sharing among those systems. High performance communication and data sharing among a large number of OS/390 systems could be technically difficult. But with the Parallel Sysplex, high performance data sharing through a coupling technology (Coupling Facility) gives high performance multisystem data sharing capability to authorized applications, such as OS/390 subsystems.

Use of the Coupling Facility by subsystems, such as Information Management System (IMS), ensures the integrity and consistency of data throughout the entire sysplex. The capability of linking together many systems and providing multisystem data sharing makes the sysplex platform ideal for parallel processing, particularly for online transaction processing (OLTP) and decision support.

In short, a parallel sysplex builds on the base sysplex capability, and allows you to increase the number of CPCs and OS/390 images that can directly share work. The Coupling Facility enables high performance, multisystem data sharing across all the systems. In addition, workloads can be dynamically balanced across systems with the help of workload management functions.



IEASYSxx Requirements

★ PLEXCFG=	
– XCFLOCAL	– local mode
– MONOPLEX	– one system sysplex
– MULTISYSTEM	– member of sysplex
– ANY	– any sysplex
★ GRS=TRYJOIN/STAR	– GRS ring or star mode
★ CLOCK=xx	– clock member
★ COUPLE=xx	– couple member
★ CON=xx	– console member
★ GRSCNF=xx	– define GRS complex
★ GRSRNL=xx	– identify the resource name list (RNL)
★ SYSNAME=xxxx	– name of the system

Figure 28. Sysplex parmlib parameters

1.10 IEASYSxx parmlib parameters

The values you specify in SYS1.PARMLIB largely control the characteristics of systems in a sysplex. Many of the values that represent the fundamental decisions you make about the systems in your sysplex are in SYS1.PARMLIB.

IEASYSxx is the system parameter list, which holds parameter values that control the initialization of OS/390. The key parmlib members you need to consider when setting up an OS/390 system to run in a sysplex are shown in the visual and are discussed in this section and the following visuals.

1.10.1 SYSNAME parmlib specification

You can specify the system name on the SYSNAME parameter in the IEASYSxx parmlib member. That name can be overridden by a SYSNAME value specified either in the IEASYMxx parmlib member or in response to the IEA101A Specify System Parameters prompt. IEASYMxx provides a single place to define system parameters and system symbols for all systems in a sysplex.

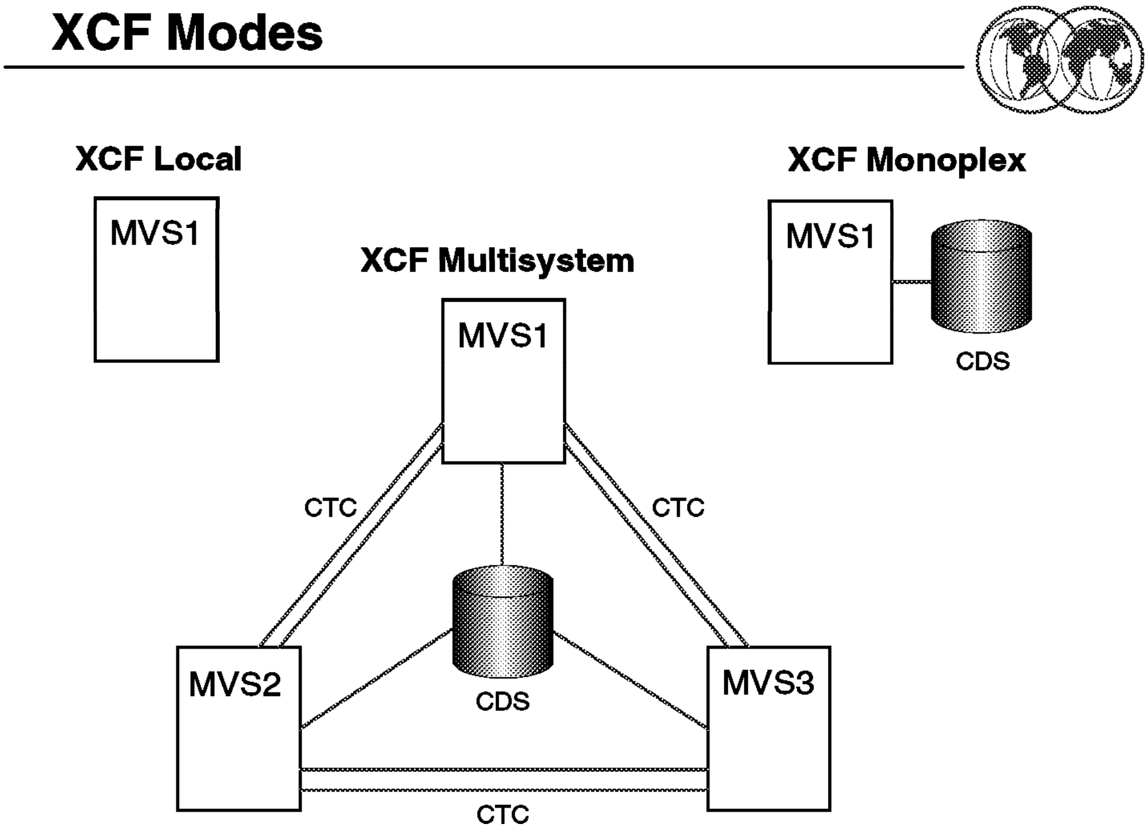


Figure 29. XCF modes

1.10.2 XCF modes

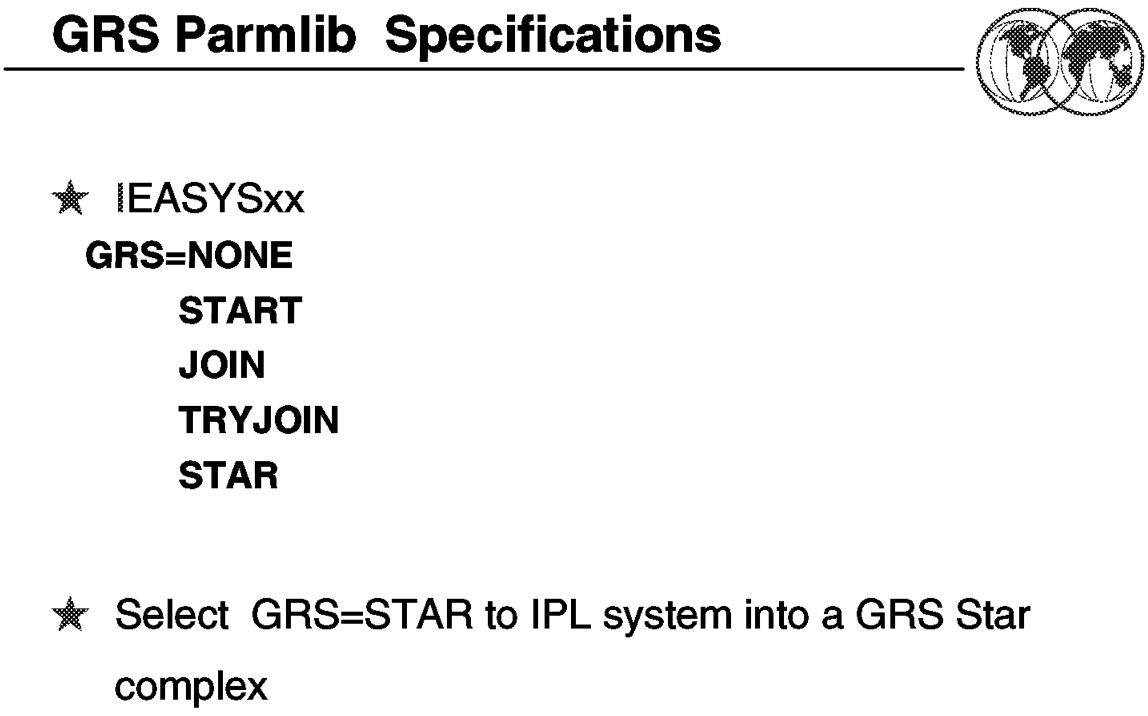
The PLEXCFG IEASYSxx parameter restricts the type of sysplex configuration into which the system is allowed to IPL. The following modes can be specified:

MULTISYSTEM PLEXCFG=MULTISYSTEM indicates that the system is to be part of a sysplex consisting of one or more OS/390 systems that reside on one or more processors. The same sysplex couple data sets must be used by all systems.

XCFLOCAL PLEXCFG=XCFLOCAL indicates that the system is to be a single, stand-alone OS/390 system that is not a member of a sysplex and cannot use couple data sets. The COUPLExx parmlib member cannot specify a sysplex couple data set, and, therefore, other couple data sets cannot be used. Thus, functions, such as WLM, that require a couple data set are not available.

MONOPLEX PLEXCFG=MONOPLEX indicates that the system is to be a single-system sysplex that must use a sysplex couple data set. Additional couple data sets, such as those that contain policy information, can also be used. XCF coupling services are available on the system, and multisystem applications can create groups and members. Messages can flow between members on this system (but not between this system and other OS/390 systems) via XCF signalling services. If signalling paths are specified, they are not used.

ANY PLEXCFG=ANY indicates that the system can be part of any valid sysplex configuration. Specifying ANY is logically equivalent to specifying XCFLOCAL, MONOPLEX, or MULTISYSTEM. ANY is the default. If the system is ever to be part of a single-system sysplex or a multisystem sysplex, you must specify a COUPLExx parmlib member that gives the system access to a couple data set.



GRS Parmlib Specifications

- IEASYSxx

  GRS=NONE
  START
  JOIN
  TRYJOIN
  STAR

- Select GRS=STAR to IPL system into a GRS Star complex

Figure 30. GRS parmlib specifications

1.10.3 GRS parmlib specifications

The GRS IEASYSxx parameter indicates whether the system is to join a global resource serialization complex.

In a multisystem sysplex, every system in the sysplex must be in the same global resource serialization complex. This allows global serialization of resources in the sysplex. To initialize each OS/390 system in the multisystem sysplex, you must use the global resource serialization component of OS/390.

Every system in a sysplex is a member of the same global resource serialization complex. Global resource serialization is required in a sysplex because components and products that use sysplex services need to access a sysplex-wide serialization mechanism. You can set up either of the following types of complex for global resource serialization:

- GRS=STAR

  In a GRS star complex, all of the systems in the sysplex must match the systems in the complex. All systems are connected to a Coupling Facility lock structure in a *star* configuration via signalling paths, or XCF communication paths, or both.

  The effect of STAR depends on the sysplex configuration.

  If PLEXCFG=XCFLOCAL or MONOPLEX the configuration is not allowed in a star complex.

  If PLEXCFG=MULTISYSTEM the system starts or joins an existing sysplex and a global resource serialization star complex. XCF coupling services are used in the sysplex and in the complex.

- GRS=TRYJOIN/START/JOIN

  In a GRS ring complex, the systems in the sysplex are the same as the systems in the complex or at least one system in the complex is outside of the sysplex. In a ring, you derive greater benefits if the sysplex and the complex match. All systems are connected to each other in a *ring* configuration via global resource serialization managed channel-to-channel (CTCs) adapters, XCF communication paths (CTCs), and XCF signalling paths through a Coupling Facility.

  The effect of JOIN, START, or TRYJOIN depends on the sysplex configuration.

  – If PLEXCFG=XCFLOCAL or MONOPLEX and there is only one system in the sysplex, the system starts or joins an existing global resource serialization complex and non-XCF protocols are used in the complex.

  – If PLEXCFG=MULTISYSTEM the system starts or joins an existing sysplex and a global resource serialization complex. XCF coupling services are used in the sysplex and the complex.

- GRS=NONE

  The effect of NONE depends on the intended sysplex configuration:

  – If PLEXCFG=XCFLOCAL or MONOPLEX and there is only one system in the sysplex and no global resource serialization complex, GRS=none is allowed.

  – If PLEXCFG=MULTISYSTEM, GRS=none is not allowed. Global resource serialization is required in a multisystem sysplex.

- GRSCNF=xx

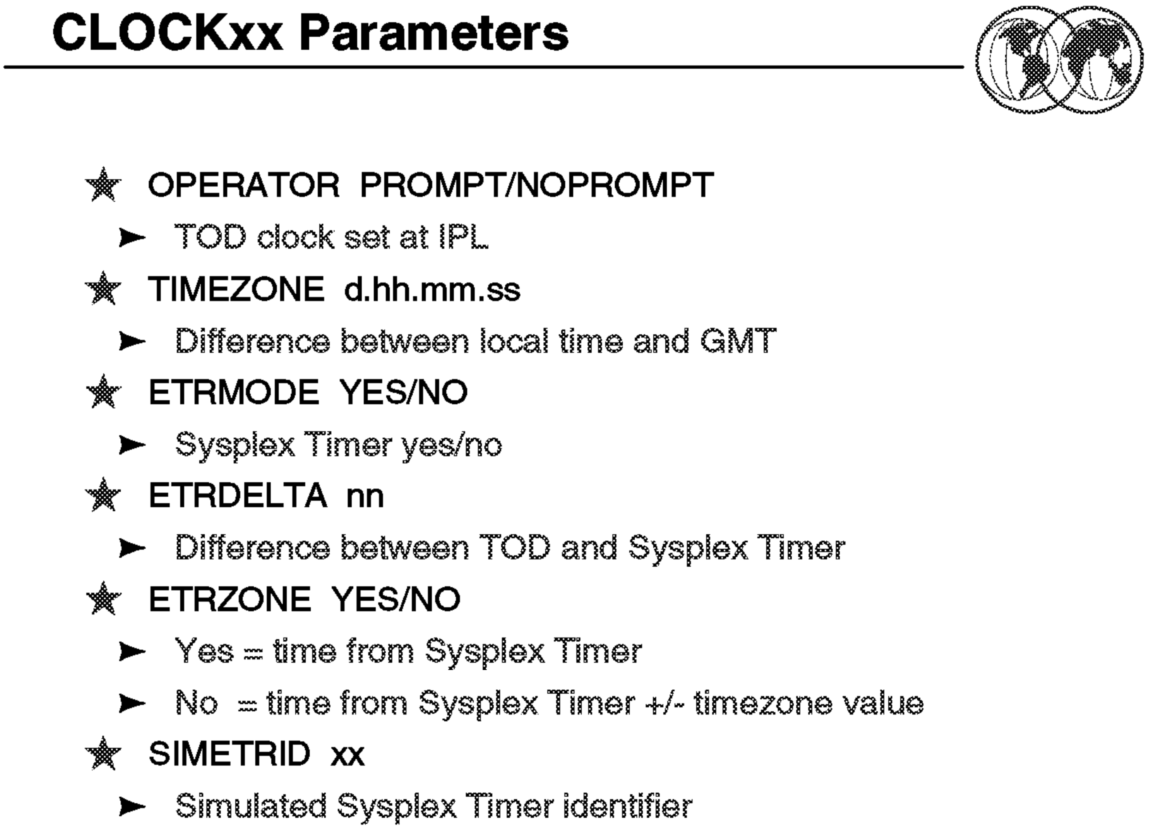
  This parameter specifies the GRSCNFxx parmlib member that describes the global resource serialization complex for the initializing system. Use GRSCNF=00 (the default) when all systems in the sysplex are in the global resource serialization ring complex and you can use the values in the default GRSCNF00 parmlib member to provide control information about the complex. With GRSCNF00, all global resource serialization communication is through XCF services.

- GRSRNLxx

  This parameter specifies the resource name lists to be used to control the serialization of resources in the complex.

  Use the GRSRNL=00 system parameter when you can use the default RNLs in the GRSRNL00 parmlib member for global resource serialization in the complex.

  Use GRSRNL=EXCLUDE when you do not have an existing global resource serialization complex, are adding the system to a sysplex, and want the serialization of most resources done with the RESERVE macro. GRSRNL=EXCLUDE indicates that all global enqueues are to be treated as local enqueues.



CLOCKxx Parameters

- OPERATOR PROMPT/NOPROMPT
  - TOD clock set at IPL
- TIMEZONE d.hh.mm.ss
  - Difference between local time and GMT
- ETRMODE YES/NO
  - Sysplex Timer yes/no
- ETRDELTA nn
  - Difference between TOD and Sysplex Timer
- ETRZONE YES/NO
  - Yes = time from Sysplex Timer
  - No = time from Sysplex Timer +/- timezone value
- SIMETRID xx
  - Simulated Sysplex Timer identifier

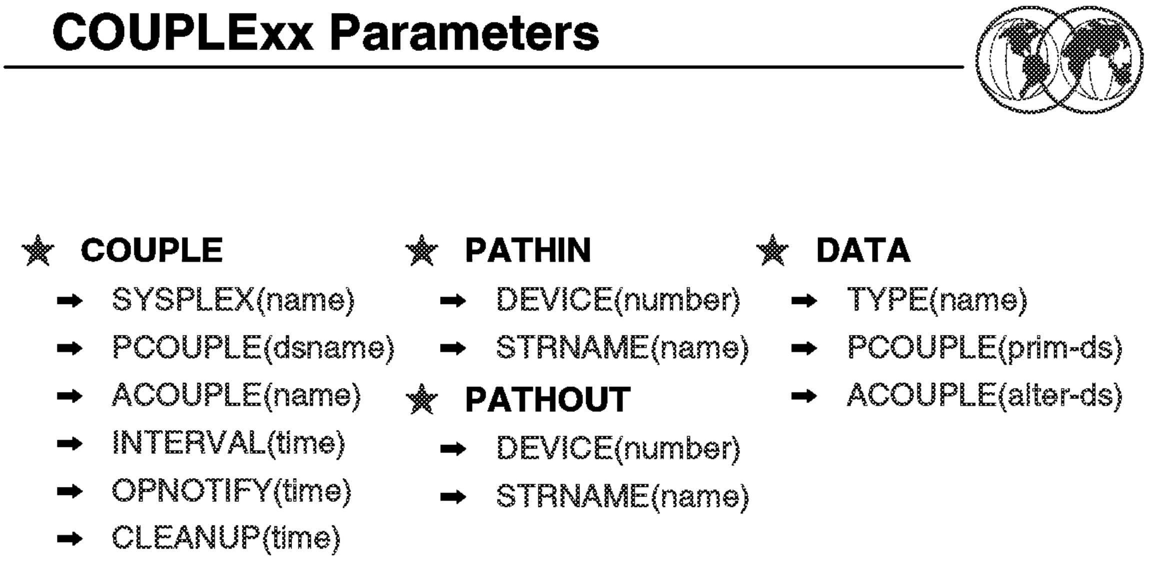
Figure 31. CLOCKxx parameters

1.10.4 CLOCKxx parameters

The CLOCK=xx parameter in the IEASYSxx parmlib member specifies the CLOCKxx parmlib member to be used when you IPL your system. CLOCKxx indicates how the time of day (TOD) is to be set on the system. OS/390 bases its decisions and activities on the assumption that time is progressing forward at a constant rate. The instrument used to mark time in an OS/390 system is the time of day (TOD) clock. As it operates, OS/390 obtains time stamps from the TOD clock. OS/390 uses these time stamps to:

- Identify the sequence of events in the system
- Determine the duration of an activity
- Record the time on online reports or printed output
- Record the time in online logs used for recovery purposes

To ensure that OS/390 makes time-related decisions as you intend, IBM recommends that you do not reset your TOD clock, for example, to switch to or from Daylight Saving Time. Instead, set the TOD clock to a standard time origin, such as Greenwich Mean Time (GMT), and use the TIMEZONE statement of the CLOCKxx parmlib member to adjust for local time. (The TIMEZONE statement allows you to specify how many hours east or west of Greenwich you are.) To adjust local time, change the offset value for the TIMEZONE statement. This will not disturb OS/390′s assumption that time is progressing forward and will allow time stamps on printed output and displays to match local time.



COUPLExx Parameters

★ **COUPLE**

- ➡ SYSPLEX(name)
- ➡ PCOUPLE(dsname)
- ➡ ACOUPLE(name)
- ➡ INTERVAL(time)
- ➡ OPNOTIFY(time)
- ➡ CLEANUP(time)

★ **PATHIN**

- ➡ DEVICE(number)
- ➡ STRNAME(name)

★ **PATHOUT**

- ➡ DEVICE(number)
- ➡ STRNAME(name)

★ **DATA**

- ➡ TYPE(name)
- ➡ PCOUPLE(prim-ds)
- ➡ ACOUPLE(alter-ds)

Figure 32. COUPLExx parameter

1.10.5 COUPLExx parameter

The COUPLE=xx parameter in the IEASYSxx parmlib member specifies the COUPLExx parmlib member to be used. It contains values that reflect fundamental decisions an installation makes as it sets up a sysplex. The COUPLExx statement includes the following keywords:

SYSPLEX SYSPLEX contains the name of the sysplex. The sysplex name allows a system to become part of the named sysplex. Specify the same sysplex-name for each OS/390 system in the sysplex. Sysplex-name must match the sysplex name specified on the couple data sets when they were formatted. The sysplex name is also the substitution text for the &SYSPLEX system symbol.

PCOUPLE PCOUPLE contains the names of the primary and sysplex couple data sets.

ACOUPLE ACOUPLE contains the names of the alternate sysplex couple data sets.

INTERVAL INTERVAL can contain values that reflect recovery-related decisions for the sysplex. INTERVAL specifies the failure detection interval—the amount of elapsed time at which XCF on another system is to initiate system failure processing for this system because XCF on this system has not updated its status within the specified time.

OPNOTIFY OPNOTIFY can contain values that reflect recovery-related decisions for the sysplex. OPNOTIFY specifies the amount of elapsed time at which XCF on another system is to notify the operator that this system has not updated its status. This value must be greater than or equal to the value specified on the INTERVAL keyword.

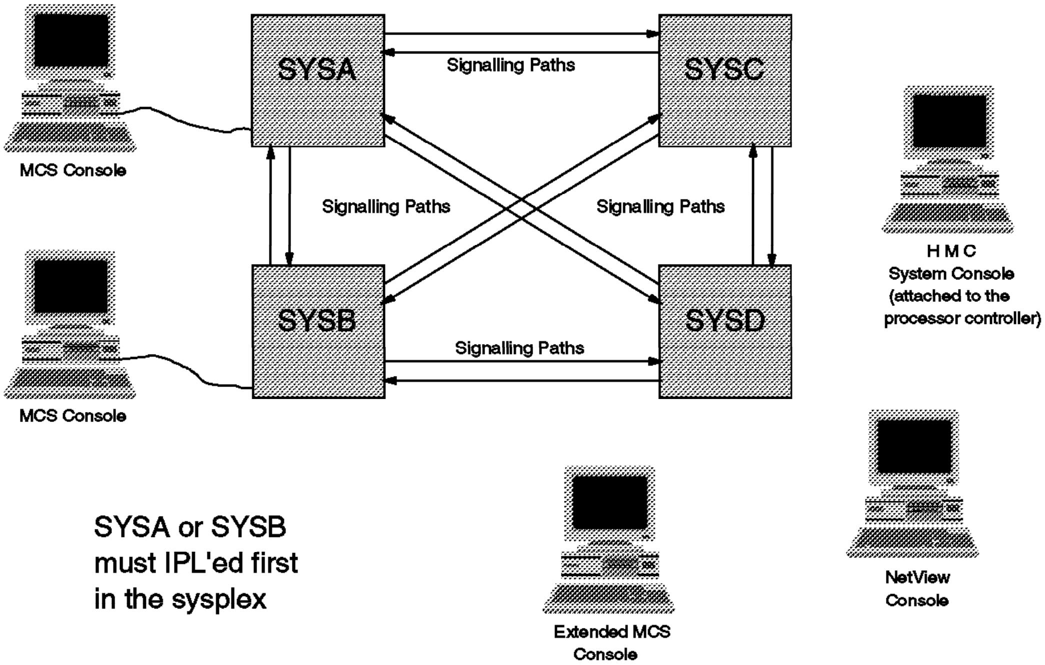
CLEANUP CLEANUP can contain values that reflect recovery-related decisions for the sysplex. CLEANUP specifies how many seconds the system waits between notifying members that this system is terminating and loading a non-restartable wait state. This is the amount of time members of the sysplex have to perform cleanup processing.

PATHIN/PATHOUT PATHIN and PATHOUT describe the XCF signalling paths for inbound/outbound message traffic. More than one PATHIN/PATHOUT statement can be specified. The PATHIN/PATHOUT statement is not required for a single-system sysplex.

- DEVICE specifies the device number(s) of a signalling path used to receive/send messages sent from another system in the sysplex.
- STRNAME specifies the name of one or more Coupling Facility list structures that are to be used to establish XCF signalling paths.

Note: Either the STRNAME keyword or the DEVICE keyword is required.

Sysplex Consoles



SYSA or SYSB
must IPL'ed first
in the sysplex

Figure 33. Sysplex consoles

1.10.6 Consoles in a sysplex

Traditionally, operators on an OS/390 image receive messages and enter commands from multisystem console support (MCS) consoles. With multisystem console support, a sysplex comprised of many OS/390 images can be operated from a single console, giving the operator a single point of control for all images.

There are three types of operator consoles in a sysplex:

- MCS consoles

  MCS consoles are display devices that are attached to an OS/390 system and provide the basic communication between operator and OS/390. MCS consoles must be locally channel-attached to non-SNA 3x74 control units; there is no MCS console support for any SNA-attached devices. You can define a maximum of 99 MCS consoles, including any subsystem allocatable consoles for an OS/390 system. In a Parallel Sysplex, the limit is also 99 consoles for the entire Parallel Sysplex, which means that you may have to consider this in your configuration planning. One possible way to alleviate this restriction is through the use of extended MCS consoles.

- Extended MCS consoles

  Programmable extended MCS consoles are defined and activated by authorized programs acting as operators. An extended MCS console is actually a program that acts as a console. It is used to issue OS/390 commands and to receive command responses, unsolicited message traffic, and the hardcopy message set. There are two ways to use extended MCS consoles:

  – Interactively, through IBM products such as TSO/E, SDSF, and NetView

  - Through user-written application programs

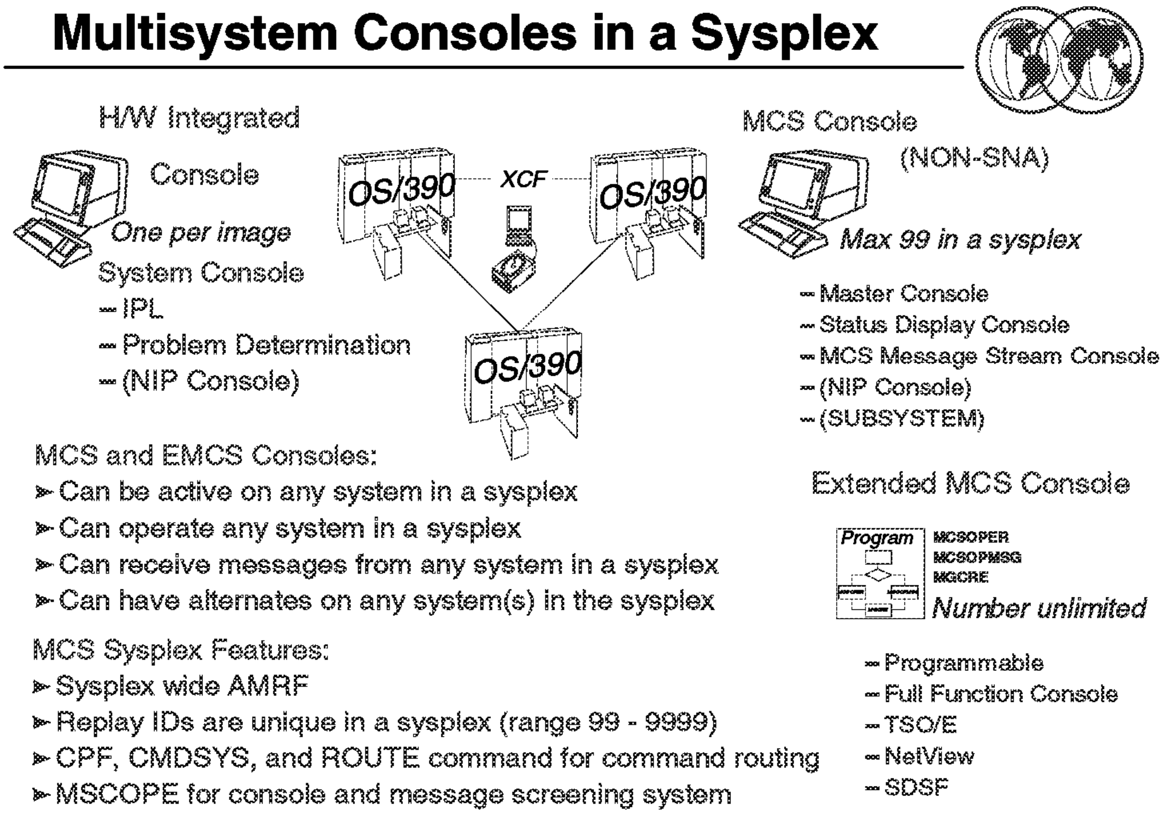
  Generally speaking, an extended MCS console is used for almost any of the functions that are performed from an MCS console. It can also be controlled in a manner that is similar to an MCS console.

- Integrated (system) consoles

  This term refers to the interface provided by the Hardware Management Console (HMC) on an IBM 9672. It is referred to as SYSCONS and does not have a device number. There are three system functions which might use this interface:

  - Nucleus Initialization Program (NIP)
  - Disabled Console Communication Facility (DCCF)
  - Multiple Console Support (MCS)

  The system console is automatically defined during OS/390 initialization.



Multisystem Consoles in a Sysplex

H/W Integrated Console

One per image

System Console

- IPL
- Problem Determination
- (NIP Console)

MCS Console (NON-SNA)

Max 99 in a sysplex

- Master Console
- Status Display Console
- MCS Message Stream Console
- (NIP Console)
- (SUBSYSTEM)

MCS and EMCS Consoles:

- Can be active on any system in a sysplex
- Can operate any system in a sysplex
- Can receive messages from any system in a sysplex
- Can have alternates on any system(s) in the sysplex

Extended MCS Console

Number unlimited

- Programmable
- Full Function Console
- TSO/E
- NetView
- SDSF

MCS Sysplex Features:

- Sysplex wide AMRF
- Replay IDs are unique in a sysplex (range 99 - 9999)
- CPF, CMDSYS, and ROUTE command for command routing
- MSCOPE for console and message screening system

Figure 34. Multisystem consoles in a sysplex

1.10.7 Multisystem consoles in a sysplex

In a Parallel Sysplex implementation, there is no requirement that you have an MCS console on every system in the Parallel Sysplex. Using command and message routing capabilities, from one MCS console or extended MCS console, it is possible to control multiple systems in the Parallel Sysplex. Although MCS consoles are not required on all systems, you should plan the configuration carefully to ensure that there is an adequate number to handle the message traffic and to provide a valid configuration, regardless of the number of systems in the Parallel Sysplex at a time.

If there is neither an MCS console nor an integrated system console on a system, that system probably cannot be the first to be IPLed into a sysplex; in addition, synchronous messages would go to the hardware system console, or a wait state would result. Alternate consoles must also be considered across the entire sysplex, especially for the sysplex master console. You should plan the console configuration so that there is always an alternate to the sysplex master console available at all times. If you do not do so, unnecessary operator action will be required, and messages may be lost or sent to the hardcopy log.

A given computing operations environment can have a variety of different types of operators and different types of consoles. This chapter focuses primarily on the OS/390 console operator′s tasks, and how the OS/390 operator′s job might be different in a sysplex.

An OS/390 console operator must start, run, and stop the OS/390 operating system. That involves controlling OS/390 system software, and the hardware it runs on, including processors, channel paths, and I/O devices. The operator might deal with messages and problems from many sources including

OS/390, JES, DFSMS/MVS, and other subsystems and products such as CICS, IMS, and VTAM, to name just a few.

Some of the major tasks of operating the OS/390 system include:

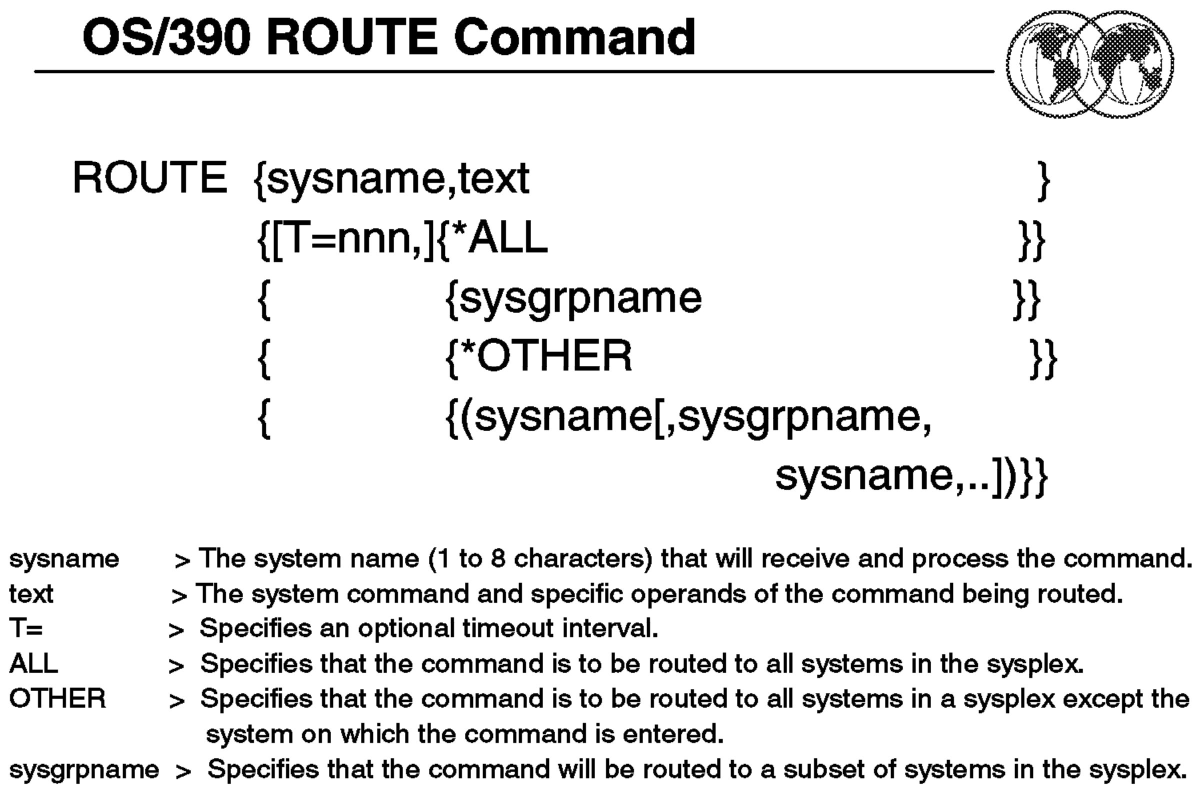
- Starting the system
- Controlling the system
- Controlling jobs
- Controlling system information recording
- Responding to failures
- Changing the configuration
- Providing problem determination data
- Quiescing the system
- Stopping the system

How does the operator who is not operating a sysplex perform tasks? Usually, the operators use multiple consoles to receive messages and issue the commands necessary to perform tasks.

In the sysplex, the major tasks of operating the system do not change very much, and you can still use consoles to receive messages and issue commands to accomplish the tasks. The potential exists, however, for the number of consoles to increase, and for the tasks to become more repetitive and more complex because of the existence of multiple images of multiple products.

The potential also exists in the sysplex for the operator to receive a great many messages, because you have the ability to route all messages from all OS/390 systems to a single console.

In the sysplex environment you want to reduce the number of consoles that operators have to deal with, reduce or eliminate the repetition, and generally simplify the picture. You should automate or suppress most messages so that operators will see and deal with fewer messages in this environment. In addition, operators will be using graphical displays in addition to 3270 displays. (This changes the way operators interact with the system to perform tasks, and might require them to have additional training to use a graphical workstation.)



OS/390 ROUTE Command

```
ROUTE {sysname,text                              }
      {[T=nnn,]{*ALL                             }}
      {        {sysgrpname                       }}
      {        {*OTHER                           }}
      {        {(sysname[,sysgrpname,
                                     sysname,..])}}
```

sysname > The system name (1 to 8 characters) that will receive and process the command.
text > The system command and specific operands of the command being routed.
T= > Specifies an optional timeout interval.
ALL > Specifies that the command is to be routed to all systems in the sysplex.
OTHER > Specifies that the command is to be routed to all systems in a sysplex except the system on which the command is entered.
sysgrpname > Specifies that the command will be routed to a subset of systems in the sysplex.

Figure 35. OS/390 ROUTE command

1.10.8 OS/390 ROUTE command

The following sections explain how the goals of single system image, single point of control, and minimal human intervention help to simplify the operator′s job in a sysplex.

1.10.8.1 Single system image

Single system image allows the operator, for certain tasks, to interact with multiple images of a product as though they were one image. For example, the operator can issue a single command to all OS/390 systems in the sysplex instead of repeating the command for each system.

The operator can use the OS/390 ROUTE command to direct a command to all systems in the sysplex (ROUTE *ALL), or a subset of the systems in the sysplex (ROUTE system_group_name). When issuing a command to multiple systems, the operator can receive a consolidated response to one console, rather than receiving responses from each system that must be located and correlated. Also, WTORs are global in the sysplex; the operator can reply to a WTOR from any console in the sysplex, even if the console is not directly attached to the system that issued the WTOR.

Some operator commands also have a sysplex scope independent of any routing or automation (for example, certain forms of DISPLAY XCF and SETXCF).

1.10.8.2 Single point of control

Single point of control allows the operator to interact with a suite of products from a single workstation, without in some cases knowing which products are performing which functions.

Remember that single point of control does not necessarily imply that you would have one single workstation from which all tasks by all people in the computing center would be done. But you could set things up so that each operator can accomplish a set of tasks from a single workstation, thereby reducing the number of consoles the operator has to deal with.

If that workstation is also a graphical workstation where tasks can be performed by using a mouse to select ("click on") choices or functions to be performed, you have also reduced the complexity of performing the tasks.

One of the assumptions of single point of control is that you can receive messages from all systems on a single system. This does not happen automatically. We recommend you set up most of your OS/390 MCS consoles to receive messages from all systems, for the following reasons:

- You might need messages from multiple systems for diagnosing problems.
- Consolidating messages reduces the number of consoles you need.

OS/390 MCS consoles are non-SNA consoles that control OS/390 operations through messages and commands. These consoles can be defined to help route messages and commands to any or all systems in the sysplex for processing. You can also define extended MCS console programs to act as automated console interfaces in the sysplex.

1.10.8.3 Entering operator commands

OS/390 operator commands can provide key data for problem analysis. The commands can be entered from the following consoles:

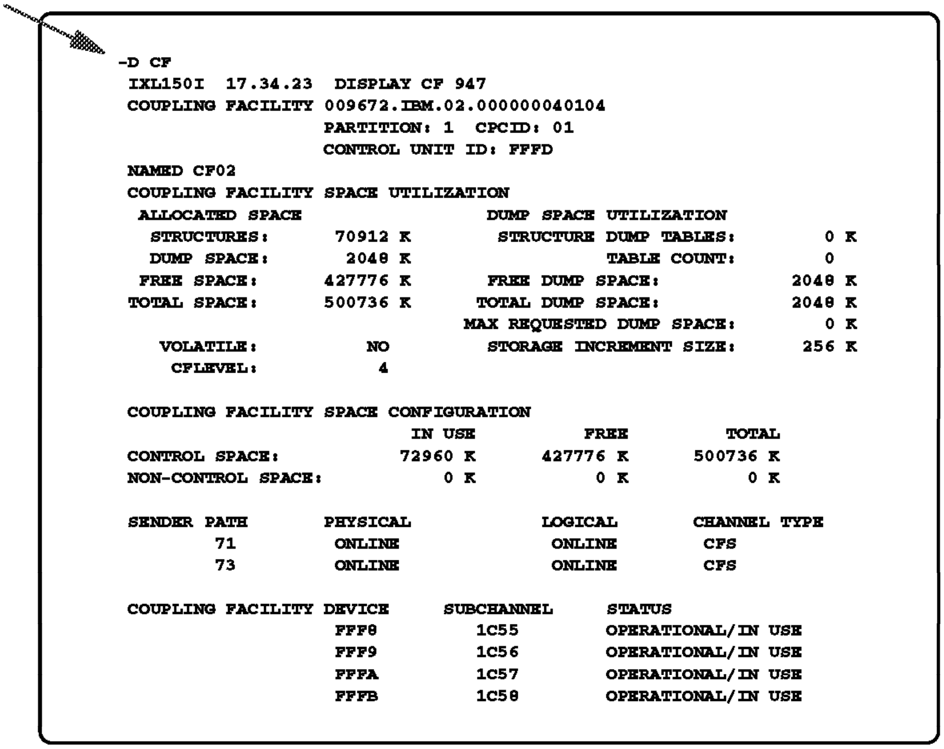
- The multiple console support (MCS) console
- The system console
- The Hardware Management Console
- The NetView console

The commands can also be issued by NetView automation CLISTs and by programs that use the extended MCS support.

Some commands have a sysplex scope, while others can be routed to systems in the sysplex with the ROUTE command. For example, the DUMP command has sysplex scope; a DUMP command entered on one system can request dumps on one, several, or all systems in a sysplex.

If a problem management tool indicates a failure, the operator can use DISPLAY commands that determine the detailed status of the failing system, job, application, device, system component, and so on. Based on the status, the operator has many commands to provide control and initiate actions; for example, the operator can enter commands to recover or isolate a failing resource.

Display CF Information



```
-D CF
 IXL150I  17.34.23  DISPLAY CF 947
 COUPLING FACILITY 009672.IBM.02.000000040104
                   PARTITION: 1  CPCID: 01
                   CONTROL UNIT ID: FFFD
 NAMED CF02
 COUPLING FACILITY SPACE UTILIZATION
  ALLOCATED SPACE                  DUMP SPACE UTILIZATION
   STRUCTURES:       70912 K        STRUCTURE DUMP TABLES:        0 K
   DUMP SPACE:        2048 K                  TABLE COUNT:        0
  FREE SPACE:       427776 K       FREE DUMP SPACE:            2048 K
 TOTAL SPACE:       500736 K      TOTAL DUMP SPACE:            2048 K
                                MAX REQUESTED DUMP SPACE:         0 K
    VOLATILE:           NO         STORAGE INCREMENT SIZE:      256 K
     CFLEVEL:            4

 COUPLING FACILITY SPACE CONFIGURATION
                            IN USE          FREE         TOTAL
 CONTROL SPACE:            72960 K      427776 K      500736 K
 NON-CONTROL SPACE:            0 K           0 K           0 K

 SENDER PATH       PHYSICAL            LOGICAL        CHANNEL TYPE
         71         ONLINE              ONLINE          CFS
         73         ONLINE              ONLINE          CFS

 COUPLING FACILITY DEVICE     SUBCHANNEL     STATUS
                    FFF8         1C55        OPERATIONAL/IN USE
                    FFF9         1C56        OPERATIONAL/IN USE
                    FFFA         1C57        OPERATIONAL/IN USE
                    FFFB         1C58        OPERATIONAL/IN USE
```

Figure 36. Display CF command output

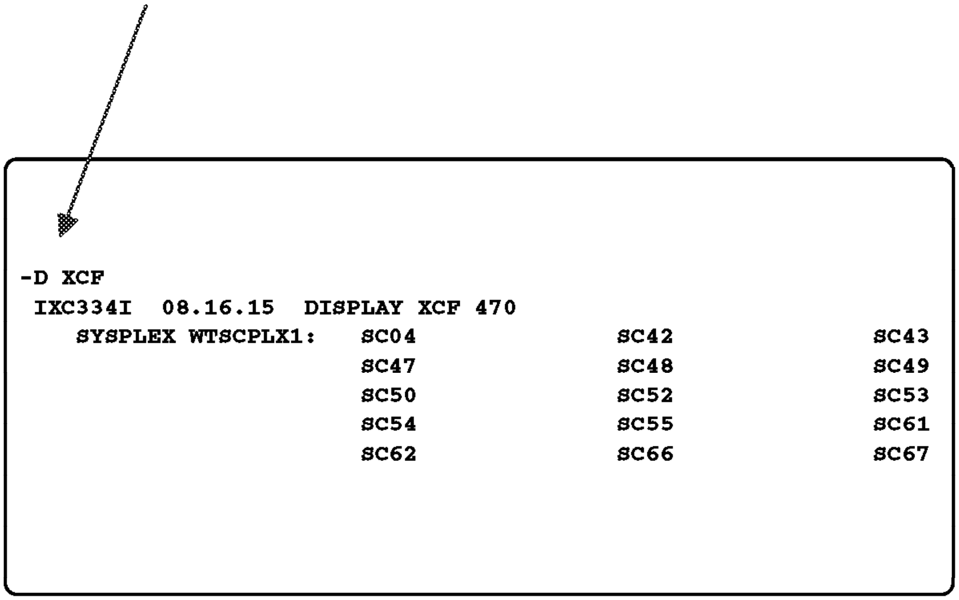
1.10.8.4 Displaying attached Coupling Facility information

Use the DISPLAY CF command to display storage and attachment information about Coupling Facilities attached to the system on which the command is processed. For example:

```
D CF,..
```

requests the system to display information about the coupling facilities that are attached to the system. If specified without further qualification, the system displays information about all Coupling Facilities that are attached.

Summary Of Current Sysplex



```
-D XCF
 IXC334I  08.16.15  DISPLAY XCF 470
    SYSPLEX WTSCPLX1:   SC04              SC42              SC43
                        SC47              SC48              SC49
                        SC50              SC52              SC53
                        SC54              SC55              SC61
                        SC62              SC66              SC67
```

Figure 37. Summary of current sysplex

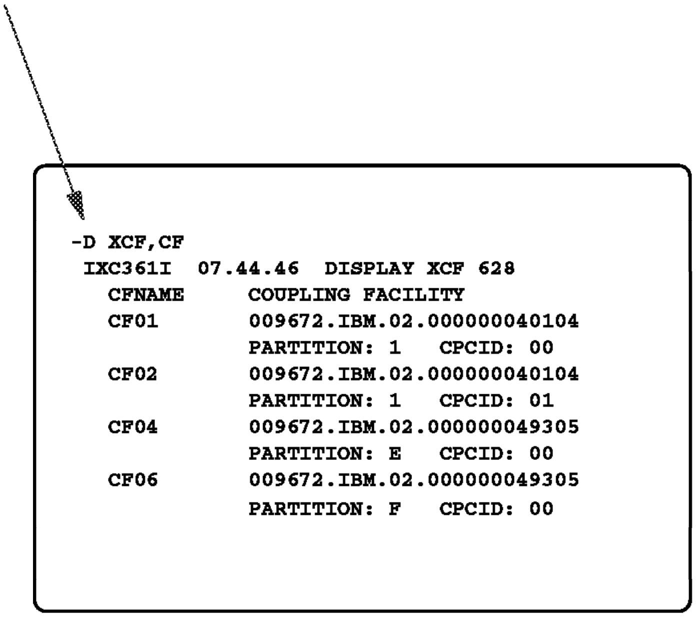
1.10.8.5 Displaying cross-system Coupling Facility (XCF) information

Use the DISPLAY XCF command to display cross system coupling information in the sysplex:

```
D XCF
```

The command displays (message IXC334I) which contains a list of all systems, by systemname, currently participating in the sysplex.

Coupling Facility Summary



```
-D XCF,CF
 IXC361I  07.44.46  DISPLAY XCF 628
   CFNAME     COUPLING FACILITY
   CF01       009672.IBM.02.000000040104
              PARTITION: 1   CPCID: 00
   CF02       009672.IBM.02.000000040104
              PARTITION: 1   CPCID: 01
   CF04       009672.IBM.02.000000049305
              PARTITION: E   CPCID: 00
   CF06       009672.IBM.02.000000049305
              PARTITION: F   CPCID: 00
```

Figure 38. Coupling Facility summary

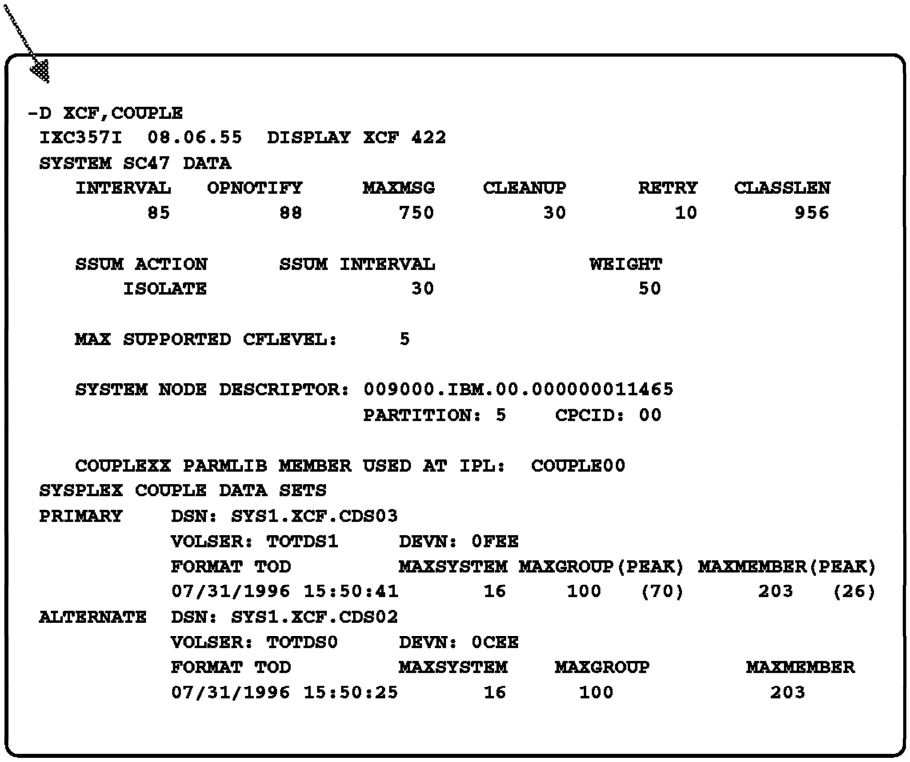
1.10.8.6 Coupling Facility summary

The following command requests information about the Coupling Facility in the policy:

```
D XCF,CF,..
```

If specified without further qualification, the system displays summary information about all Coupling Facilities that are in the policy.

Couple Data Set Information



```
-D XCF,COUPLE
 IXC357I  08.06.55  DISPLAY XCF 422
 SYSTEM SC47 DATA
    INTERVAL   OPNOTIFY     MAXMSG    CLEANUP      RETRY   CLASSLEN
          85         88        750         30         10        956

    SSUM ACTION      SSUM INTERVAL             WEIGHT
        ISOLATE                 30                 50

    MAX SUPPORTED CFLEVEL:     5

    SYSTEM NODE DESCRIPTOR: 009000.IBM.00.000000011465
                            PARTITION: 5    CPCID: 00

    COUPLEXX PARMLIB MEMBER USED AT IPL:  COUPLE00
 SYSPLEX COUPLE DATA SETS
 PRIMARY    DSN: SYS1.XCF.CDS03
            VOLSER: TOTDS1     DEVN: 0FEE
            FORMAT TOD         MAXSYSTEM MAXGROUP(PEAK) MAXMEMBER(PEAK)
            07/31/1996 15:50:41       16     100   (70)      203   (26)
 ALTERNATE  DSN: SYS1.XCF.CDS02
            VOLSER: TOTDS0     DEVN: 0CEE
            FORMAT TOD         MAXSYSTEM    MAXGROUP        MAXMEMBER
            07/31/1996 15:50:25       16      100             203
```

Figure 39. Couple data set information

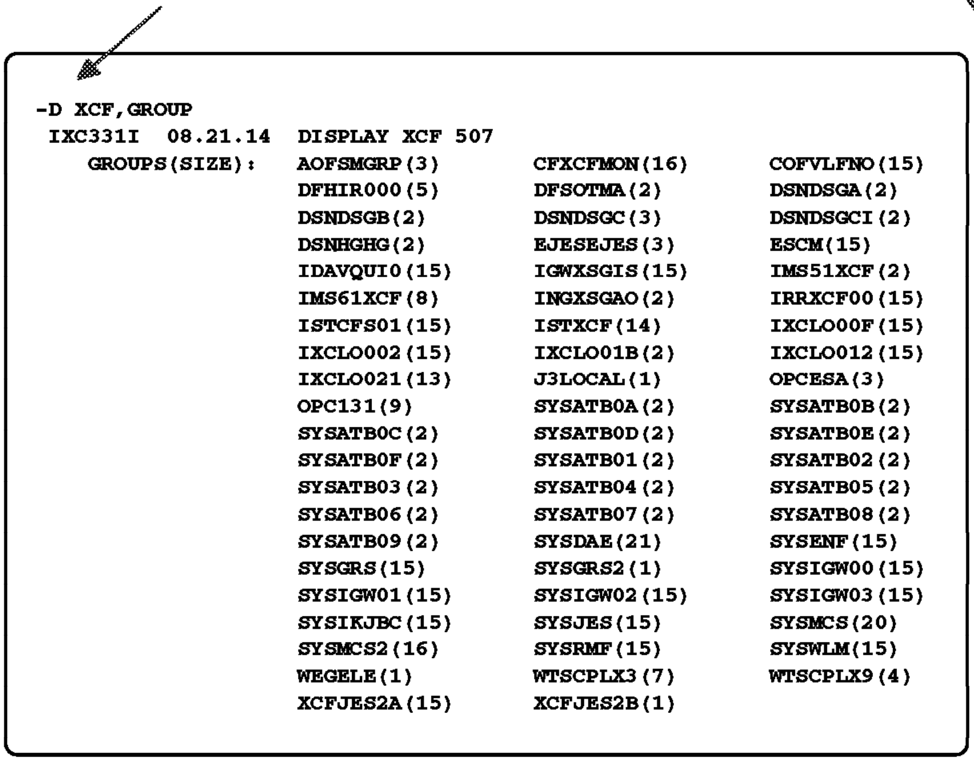
1.10.8.7 Couple data set information

The following command displays information about the couple data set in use by the sysplex:

```
D XCF,COUPLE,..
```

If specified without further qualification, information will be displayed about all couple data sets.

XCF Group List



```
-D XCF,GROUP
 IXC331I  08.21.14  DISPLAY XCF 507
    GROUPS(SIZE):   AOFSMGRP(3)       CFXCFMON(16)      COFVLFNO(15)
                    DFHIR000(5)       DFSOTMA(2)        DSNDSGA(2)
                    DSNDSGB(2)        DSNDSGC(3)        DSNDSGCI(2)
                    DSNHGHG(2)        EJESEJES(3)       ESCM(15)
                    IDAVQUI0(15)      IGWXSGIS(15)      IMS51XCF(2)
                    IMS61XCF(8)       INGXSGAO(2)       IRRXCF00(15)
                    ISTCFS01(15)      ISTXCF(14)        IXCLO00F(15)
                    IXCLO002(15)      IXCLO01B(2)       IXCLO012(15)
                    IXCLO021(13)      J3LOCAL(1)        OPCESA(3)
                    OPC131(9)         SYSATB0A(2)       SYSATB0B(2)
                    SYSATB0C(2)       SYSATB0D(2)       SYSATB0E(2)
                    SYSATB0F(2)       SYSATB01(2)       SYSATB02(2)
                    SYSATB03(2)       SYSATB04(2)       SYSATB05(2)
                    SYSATB06(2)       SYSATB07(2)       SYSATB08(2)
                    SYSATB09(2)       SYSDAE(21)        SYSENF(15)
                    SYSGRS(15)        SYSGRS2(1)        SYSIGW00(15)
                    SYSIGW01(15)      SYSIGW02(15)      SYSIGW03(15)
                    SYSIKJBC(15)      SYSJES(15)        SYSMCS(20)
                    SYSMCS2(16)       SYSRMF(15)        SYSWLM(15)
                    WEGELE(1)         WTSCPLX3(7)       WTSCPLX9(4)
                    XCFJES2A(15)      XCFJES2B(1)
```

Figure 40. XCF group list

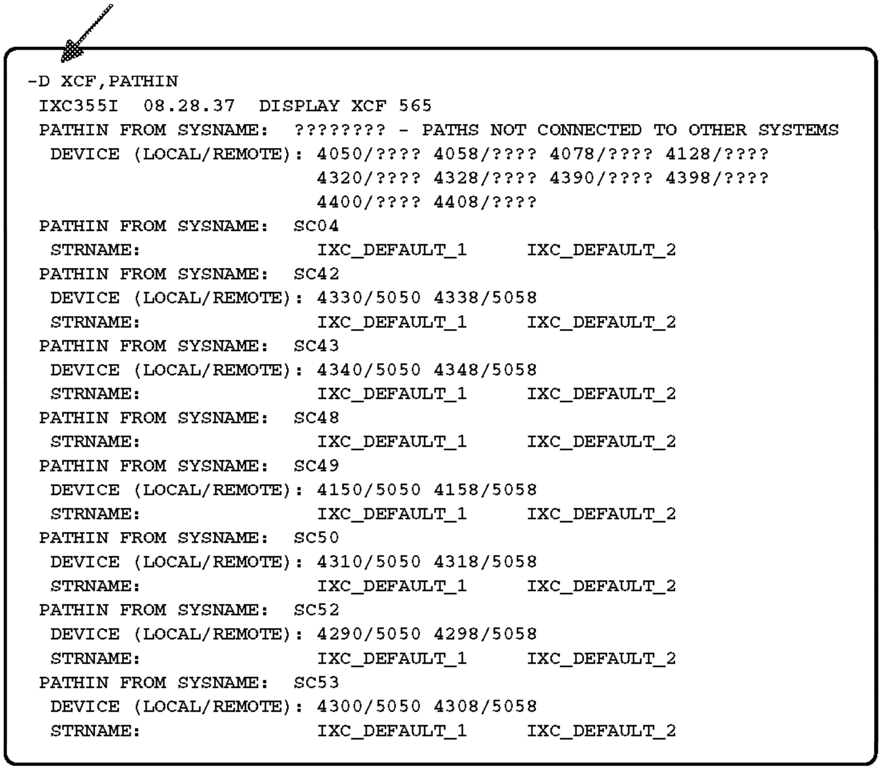
1.10.8.8 XCF group list

The following command displays information about multisystem groups:

```
D XCF,GROUP,..
```

If you do not provide a qualifying operand, a list of currently defined XCF groups is displayed.

XCF Pathin Information



```
-D XCF,PATHIN
 IXC355I  08.28.37  DISPLAY XCF 565
 PATHIN FROM SYSNAME:  ???????? - PATHS NOT CONNECTED TO OTHER SYSTEMS
  DEVICE (LOCAL/REMOTE): 4050/???? 4058/???? 4078/???? 4128/????
                         4320/???? 4328/???? 4390/???? 4398/????
                         4400/???? 4408/????
 PATHIN FROM SYSNAME:  SC04
  STRNAME:               IXC_DEFAULT_1     IXC_DEFAULT_2
 PATHIN FROM SYSNAME:  SC42
  DEVICE (LOCAL/REMOTE): 4330/5050 4338/5058
  STRNAME:               IXC_DEFAULT_1     IXC_DEFAULT_2
 PATHIN FROM SYSNAME:  SC43
  DEVICE (LOCAL/REMOTE): 4340/5050 4348/5058
  STRNAME:               IXC_DEFAULT_1     IXC_DEFAULT_2
 PATHIN FROM SYSNAME:  SC48
  STRNAME:               IXC_DEFAULT_1     IXC_DEFAULT_2
 PATHIN FROM SYSNAME:  SC49
  DEVICE (LOCAL/REMOTE): 4150/5050 4158/5058
  STRNAME:               IXC_DEFAULT_1     IXC_DEFAULT_2
 PATHIN FROM SYSNAME:  SC50
  DEVICE (LOCAL/REMOTE): 4310/5050 4318/5058
  STRNAME:               IXC_DEFAULT_1     IXC_DEFAULT_2
 PATHIN FROM SYSNAME:  SC52
  DEVICE (LOCAL/REMOTE): 4290/5050 4298/5058
  STRNAME:               IXC_DEFAULT_1     IXC_DEFAULT_2
 PATHIN FROM SYSNAME:  SC53
  DEVICE (LOCAL/REMOTE): 4300/5050 4308/5058
  STRNAME:               IXC_DEFAULT_1     IXC_DEFAULT_2
```

Figure 41. XCF path information

1.10.8.9 XCF path information

The following command displays the device number of one or more inbound signalling paths that XCF can use and information about inbound XCF signalling paths to this system:

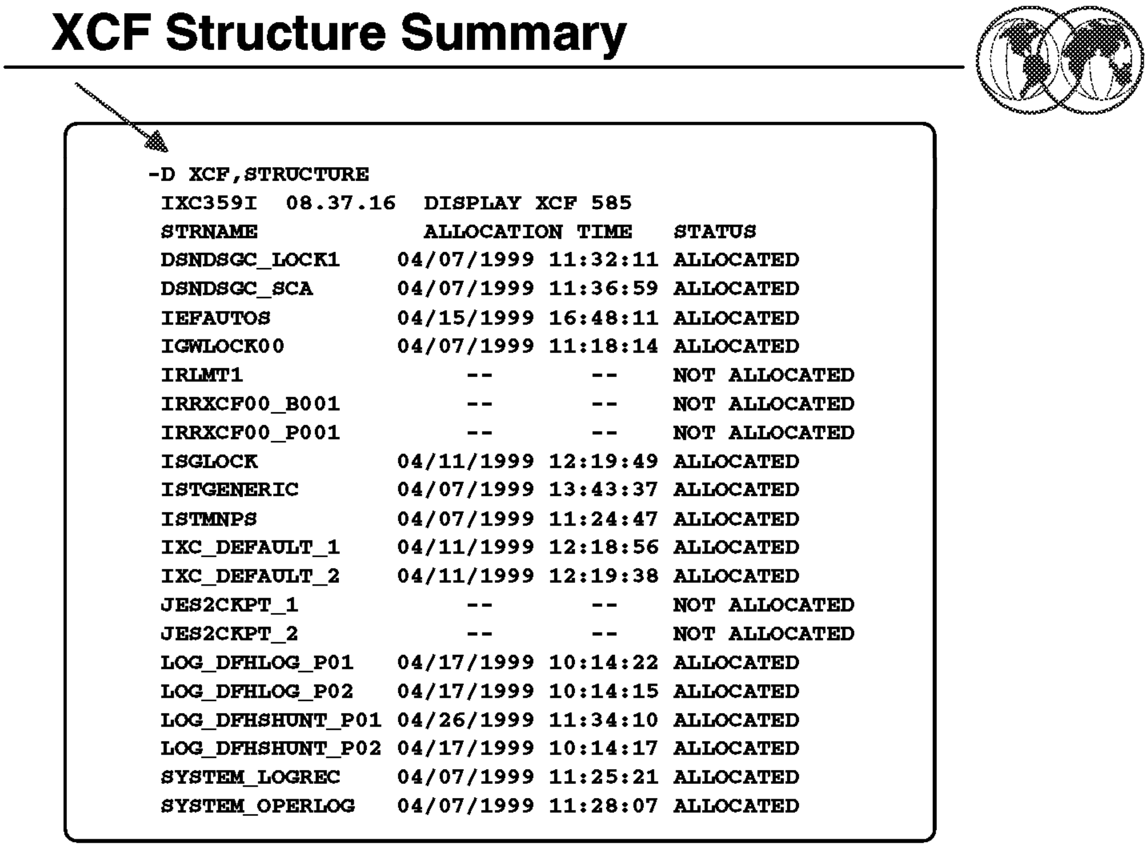
```
D XCF,PATHIN,..
```

The following command displays the device number of one or more outbound signalling paths that XCF can use, and information about outbound XCF signalling paths to this system:

```
D XCF,PATHOUT,..
```

The following command displays a list of all systems currently participating in the sysplex:

```
D XCF,SYSPLEX,..
```



XCF Structure Summary

```
-D XCF,STRUCTURE
 IXC359I  08.37.16  DISPLAY XCF 585
 STRNAME            ALLOCATION TIME   STATUS
 DSNDSGC_LOCK1    04/07/1999 11:32:11 ALLOCATED
 DSNDSGC_SCA      04/07/1999 11:36:59 ALLOCATED
 IEFAUTOS         04/15/1999 16:48:11 ALLOCATED
 IGWLOCK00        04/07/1999 11:18:14 ALLOCATED
 IRLMT1                --        --   NOT ALLOCATED
 IRRXCF00_B001         --        --   NOT ALLOCATED
 IRRXCF00_P001         --        --   NOT ALLOCATED
 ISGLOCK          04/11/1999 12:19:49 ALLOCATED
 ISTGENERIC       04/07/1999 13:43:37 ALLOCATED
 ISTMNPS          04/07/1999 11:24:47 ALLOCATED
 IXC_DEFAULT_1    04/11/1999 12:18:56 ALLOCATED
 IXC_DEFAULT_2    04/11/1999 12:19:38 ALLOCATED
 JES2CKPT_1            --        --   NOT ALLOCATED
 JES2CKPT_2            --        --   NOT ALLOCATED
 LOG_DFHLOG_P01   04/17/1999 10:14:22 ALLOCATED
 LOG_DFHLOG_P02   04/17/1999 10:14:15 ALLOCATED
 LOG_DFHSHUNT_P01 04/26/1999 11:34:10 ALLOCATED
 LOG_DFHSHUNT_P02 04/17/1999 10:14:17 ALLOCATED
 SYSTEM_LOGREC    04/07/1999 11:25:21 ALLOCATED
 SYSTEM_OPERLOG   04/07/1999 11:28:07 ALLOCATED
```

Figure 42. XCF structure summary

1.10.8.10 XCF structure summary

This command requests information about the Coupling Facility structures in the policy:

```
D XCF,STRUCTURE,..
```

If specified without further qualification, summary information will be displayed about all coupling facility structures that are in the policy.

The are other options that can be used.

1.10.8.11 XCF commands

Use the SETXCF command to control the cross-system coupling facility (XCF):

```
SETXCF COUPLE,..
```

This command can be used with options to:

- Switch a current alternate couple data set to a primary couple data set. The switch can be for either sysplex couple data sets or other types of couple data sets.
- Specify a primary non-sysplex couple data set, such as CFRM, SFM, WLM.
- Specify an alternate couple data set.
- Change options specified in the COUPLExx parmlib member.

Use the SETXCF FORCE command to clean up resources related to structures in a Coupling Facility:

```
SETXCF FORCE,..
```

The resources can be either structures actively in use in the sysplex or dumps associated with structures pending deallocation.

Use the SETXCF MODIFY command to change current XCF parameters:

```
SETXCF MODIFY,..
```

The system changes only those parameters explicitly provided on the SETXCF MODIFY command; all other parameters associated with the resource remain the same.

Use the SETXCF PRSMPOLICY command to either activate an XCF PR/SM policy, or deactivate a current active XCF PR/SM policy:

```
SETXCF PRSMPOLICY,..
```

Use the SETXCF START command:

```
SETXCF START,..
```

to do the following:

- Start new inbound signalling paths or restart inoperative inbound signalling paths.
- Start outbound signalling paths or restart inoperative outbound signalling paths.
- Define transport classes.
- Start using a new administrative policy as an active policy.
- Start rebuilding one or more Coupling Facility structures either in the same Coupling Facility or in another Coupling Facility.
- Start populating a Coupling Facility that has been newly brought into service or returned to service in a sysplex with structures selected from the set of those defined in the active CFRM policy. The structures selected are those that list the Coupling Facility to be populated as higher in the structure′s preference list than the Coupling Facility in which the structure already is allocated.
- Start user-managed duplexing of one or more structures in a Coupling Facility into another Coupling Facility.
- Start altering the size of a Coupling Facility structure.

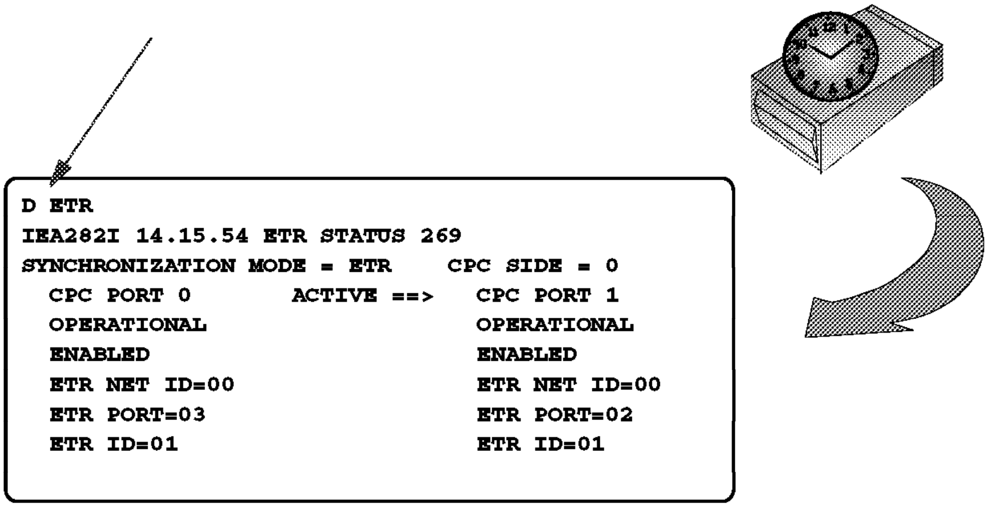
Use the SETXCF STOP command to:

```
SETXCF STOP,..
```

do the following:

- Stop one or more inbound signalling paths.
- Stop one or more outbound signalling paths.
- Delete the definition of a transport class.
- Stop using an administrative policy.
- Stop rebuilding one or more Coupling Facility structures.
- Stop populating a Coupling Facility that had been newly brought into service in a sysplex with structures selected from the set of those defined in the active CFRM policy.
- Stop user-managed duplexing of one or more structures in a Coupling Facility and specify the structure that is to remain in use.
- Stop altering a Coupling Facility structure.

Display ETR



```
D ETR
IEA282I 14.15.54 ETR STATUS 269
SYNCHRONIZATION MODE = ETR    CPC SIDE = 0
  CPC PORT 0       ACTIVE ==>   CPC PORT 1
  OPERATIONAL                   OPERATIONAL
  ENABLED                       ENABLED
  ETR NET ID=00                 ETR NET ID=00
  ETR PORT=03                   ETR PORT=02
  ETR ID=01                     ETR ID=01
```

Figure 43. Display ETR

1.10.8.12 Managing external timer

- Displaying the timer synchronization mode and ETR ports

  Use the DISPLAY ETR command to display the current timer synchronization mode and the status of the ETR ports as seen by OS/390.

  The complete syntax for the DISPLAY ETR command is:

  ```
  D ETR,DATA
  ```

  DATA is the default so you can use D ETR to display the current ETR (external time reference) synchronization and status, in detail, of each ETR port, giving the ETR network ID, ETR port number, and the ETR ID.

- Modifing the ETR

  Use the SETETR command to enable external time reference (ETR) ports that have been disabled. An ETR port disabled by a hardware problem can be enabled after the problem has been corrected.

  ```
  SETETR PORT=nn
  ```

  Port=nn specifies the number of the ETR port to be enabled. The valid values for n are 0 and 1.

- Controlling the recording of hard machine check interruptions

  You can use the MODE command to control the recording or monitoring of hard machine-check interruptions.

```
MODE AD,..

MODE SC,..
```

The AD parameter defines machine checks indicating the ETR attachment is to be monitored in the specified mode.

The SC parameter defines machine checks indicating the ETR synchronization checks are to be monitored in the specified mode.

1.10.8.13 Managing XCF images

- Removing a system from the XCF sysplex

  Use the following form of the VARY command to remove a system from the XCF sysplex.

  ```
  V XCF,systemname,OFFLINE,RETAIN={YES|NO},FORCE
  ```

  XCF,systemname,OFFLINE or OFF specifies the name of a system that XCF is to remove from the sysplex. The system that is removed is put into a wait state.

  RETAIN=YES or NO indicates whether or not XCF, on the remaining systems in the sysplex, is to retain the signalling path resources used to communicate with the system that′s removed.

  FORCE indicates that XCF will immediately remove the specified system from the sysplex. The FORCE option is only accepted after XCF has failed to remove the system with the VARY command. The VARY command with the FORCE option must be issued on the same OS/390 image where the original VARY command was issued.

- To stop a system

  Use the QUIESCE command when you want to put the system in a manual state without affecting job step timing; for example, when you want to alter storage. You can enter QUIESCE only from a console with MASTER authority. You can restart the system by performing the RESTART function.

  ```
  QUIESCE
  ```

  If possible, all jobs currently processing terminate normally. Otherwise, current activity is suspended, and the system enters a manual state or a wait state with a code of X′80000CCC′. You might receive the following message on the master console or its alternate:

  ```
  BLW002I  SYSTEM WAIT STATE ′CCC′X -- QUIESCE FUNCTION PERFORMED
  ```

Chapter 2. System logger

In a Parallel Sysplex, each system has its own SYSLOG and logrec data sets. With multiple systems, you want to merge the logs from all systems so that you have a comprehensive view of all log data within the Parallel Sysplex. MVS provides the system logger, which is a set of services that allow you to manage log data across the systems in the Parallel Sysplex. The log data is in a log stream that resides on a coupling facility; as the coupling facility fills, older data might be off-loaded to archival log stream data sets on DASD.

System logger has its own required couple data set, LOGR, that contains information to help you manage system logger resources.

System logger is a set of services that allows an application to write, browse, and delete log data. You can use system logger services to merge data from multiple instances of an application, including merging data from different systems across a sysplex.

For example, suppose you are concurrently running multiple instances of an application in a sysplex, and each application instance can update a common database. It is important for your installation to maintain a common log of all updates to the database from across the sysplex, so that if the database should be damaged, it can be restored from the backup copy. You can merge the log data from applications across the sysplex into a log stream, which is simply a collection of data in log blocks residing in the coupling facility and on DASD.

A log stream is an application-specific collection of data that is used as a log. The data is written to and read from the log stream by one or more instances of the application associated with the log stream. A log stream can be used for such purposes as a transaction log, a log for recreating databases, a recovery log, or other logs needed by applications.

A system logger application can write log data into a log stream, which is simply a collection of data. Data in a log stream spans two kinds of storage:

- Interim storage, where data can be accessed quickly without incurring DASD I/O.
- DASD log data set storage, where data is hardened for longer term access. When the interim storage medium for a log stream reaches a user-defined threshold, the log data is off-loaded to DASD log data sets.

There are two types of log streams; coupling facility log streams and DASD-only log streams. The main difference between the two types of log streams is the storage medium system logger uses to hold interim log data:

- In a coupling facility log stream, interim storage for log data is in coupling facility list structures.
- In a DASD-only log stream interim storage for log data is contained in local storage buffers on the system. Local storage buffers are data space areas associated with the system logger address space, IXGLOGR.

An installation can use just coupling facility log streams, just DASD-only log streams, or a combination of both types of log streams.

MVS System Logger

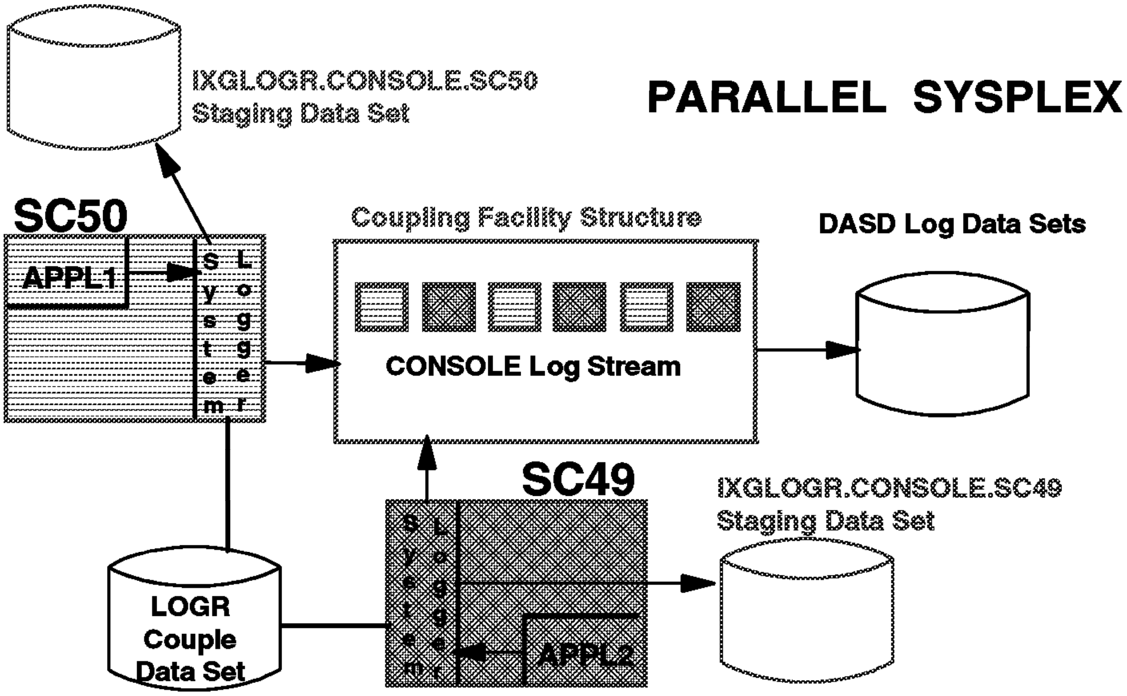


Figure 44. MVS system logger

2.1 MVS system logger

System logger is an MVS component that allows an application to log data from a sysplex environment. Data can be logged from one system or from multiple systems across the sysplex.

The OS/390 system logger is a set of services that allows an application to write, browse, and delete log data. You can merge the log data from applications in a sysplex into a log stream, which is simply a collection of data in log blocks residing in the Coupling Facility. Log blocks in the Coupling Facility can be backed up either in storage in each system or on DASD in staging data sets. When the log blocks in the Coupling Facility reach an installation-defined threshold value, they are offloaded to DASD log data sets. Therefore, at any point in time the log stream consists of records on the DASD log data sets and the log blocks currently in the Coupling Facility.

An OS/390 system logger configuration includes the system logger address space in each system, the LOGR couple data set, a log stream structure in a Coupling Facility, DASD log data sets for offloaded data from the Coupling Facility log stream, and optionally staging data sets for a backup copy of the log blocks resident in the log stream structure. All these components are shown in the visual.

The system logger address space provides the application with services and connections to the Coupling Facility. An installation must plan and predefine the Coupling Facility structures, format and define a policy for the LOGR couple data set, define the DASD log data sets, and optionally create staging data sets. An application can then issue system logger services.

A log stream is a collection of one or more log records (also referred to as log blocks) written by an application using services provided by the MVS system logger. The application using MVS system logger services may or may not have multiple instances of itself executing in a sysplex. In the case of a multi-instance application where each instance of the application writes log blocks to the same log stream, the result is a sysplex-wide merged log stream.

You can use system logger services to merge data from different systems in a sysplex, as shown in the visual where APPL1 and APPL2 are multiple instances of an application writing log blocks to the coupling facility into the CONSOLE log stream.

Every time a system logger application writes a log block to a log stream, system logger automatically makes a duplicate copy of the data as insurance against data loss due to Coupling Facility failure, system failure, or system logger failure. System logger keeps a duplex copy of data in log stream interim storage only. The duplicate copy is kept until the data is offloaded from interim storage to DASD log data sets.

The way system logger duplexes log data depends on whether the log stream is a Coupling Facility or a DASD-only log stream.

System logger allows the installation to choose between two methods of duplexing Coupling Facility log data for a Coupling Facility log stream:

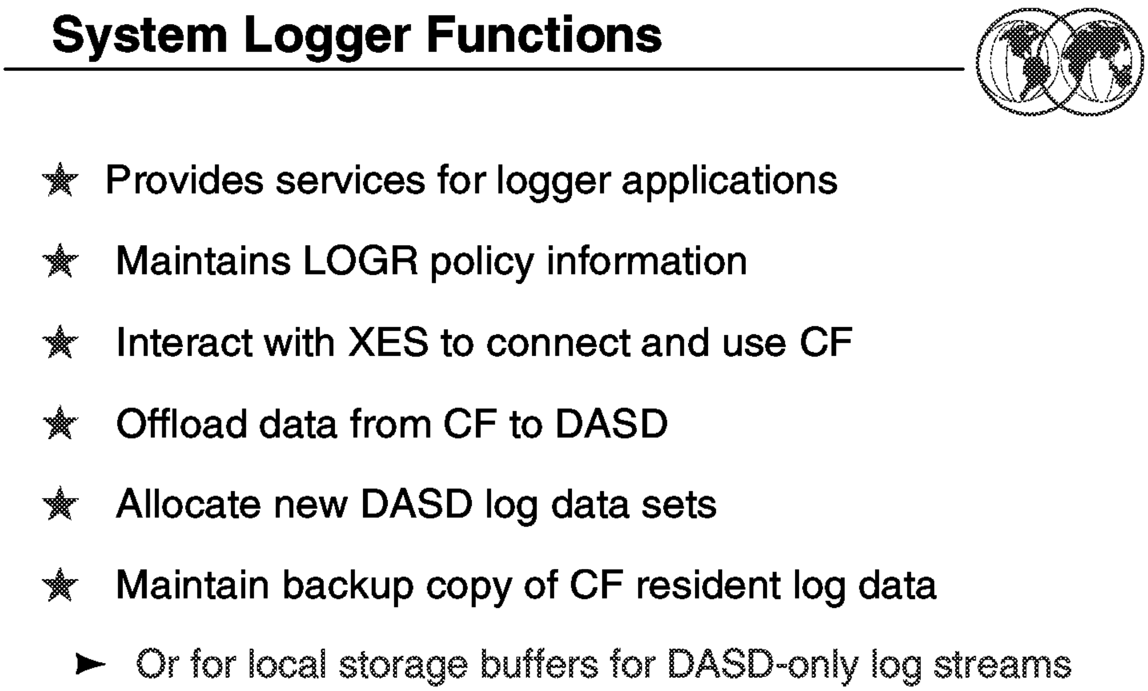
- Maintain a copy of Coupling Facility resident log data in local storage buffers on each system.

  This option is the default, and occurs automatically if you do not specifically request duplexing on staging data sets in the log stream definition in the LOGR policy.

- Maintain a copy of Coupling Facility resident log data in DASD staging data sets, one per log stream on a system.

  This option provides hardening of the data on failure-independent, persistent media.

You can request the duplexing options for a Coupling Facility log stream in the log stream definition in the LOGR policy.



System Logger Functions

- Provides services for logger applications
- Maintains LOGR policy information
- Interact with XES to connect and use CF
- Offload data from CF to DASD
- Allocate new DASD log data sets
- Maintain backup copy of CF resident log data
  - Or for local storage buffers for DASD-only log streams

Figure 45. System logger functions

2.1.1 System logger functions

System logger is a set of services that allows an application to write, browse, and delete log data. You can use system logger services to merge data from multiple instances of an application, including merging data from different systems across a sysplex.

The system logger component manages log streams based on the policy information that installations place in the LOGR couple data set. The LOGR couple data set must:

- Be accessible by all the systems in the sysplex.
- Be formatted using the IXCL1DSU utility.
- Contain policy information, which is defined using the IXCMIAPU utility or the IXGINVNT service, see *OS/390 MVS Programming: Assembler Services Guide*, GC28-1762, for more information.

When a system logger application writes a log block to a coupling facility log stream, system logger writes it first to a Coupling Facility list structure. For Coupling Facility log streams, system logger interacts with cross-system extended services (XES) to connect to and use the Coupling Facility for system logger applications. System logger requires that a Coupling Facility list structure be associated with each log stream.

When the Coupling Facility structure space allocated for the log stream reaches the installation-defined threshold, system logger moves (off-loads) the log blocks from the Coupling Facility structure to VSAM linear DASD data sets, so that the Coupling Facility space for the log stream can be used to hold new log blocks. From a user′s point of view, the actual location of the log data in the log stream is transparent.

For DASD-only log streams, system logger:

- Off-loads log data from the local storage buffers to DASD log data sets as the DASD staging data set space reaches the installation-defined thresholds.
- Automatically allocates new DASD log data sets for log streams.
- Maintains a backup copy of (duplexes) log data that is in interim storage for recovery. Log data in interim storage is vulnerable to loss due to system or sysplex failure because it has not yet been hardened to DASD log data sets. System logger duplexes interim storage log data for both Coupling Facility and DASD-only log streams.

System Logger with Coupling Facility

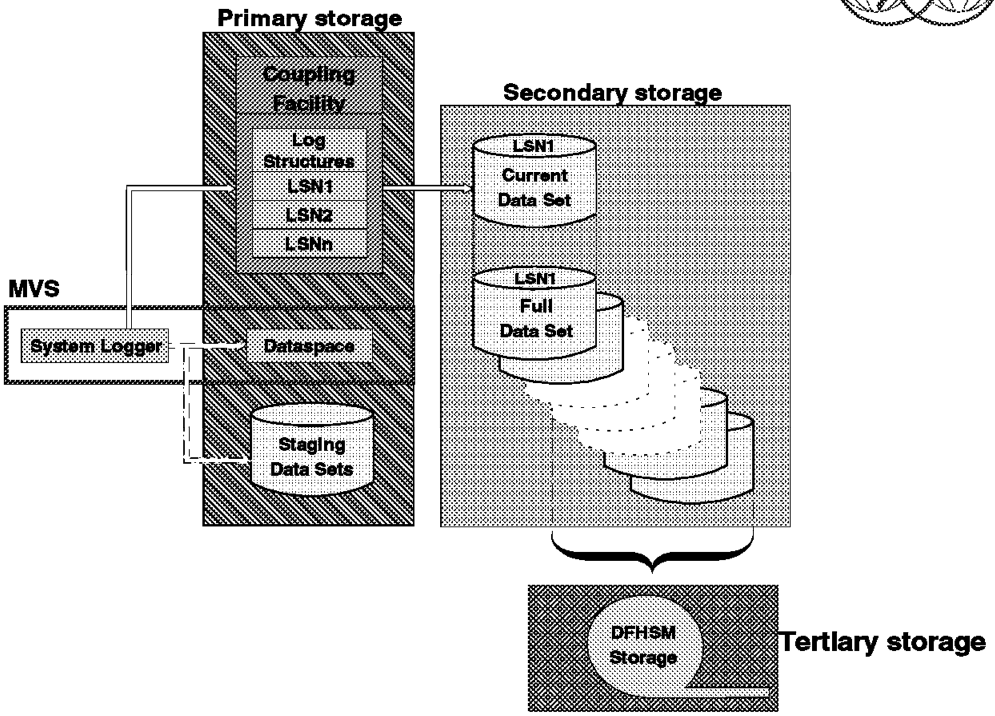
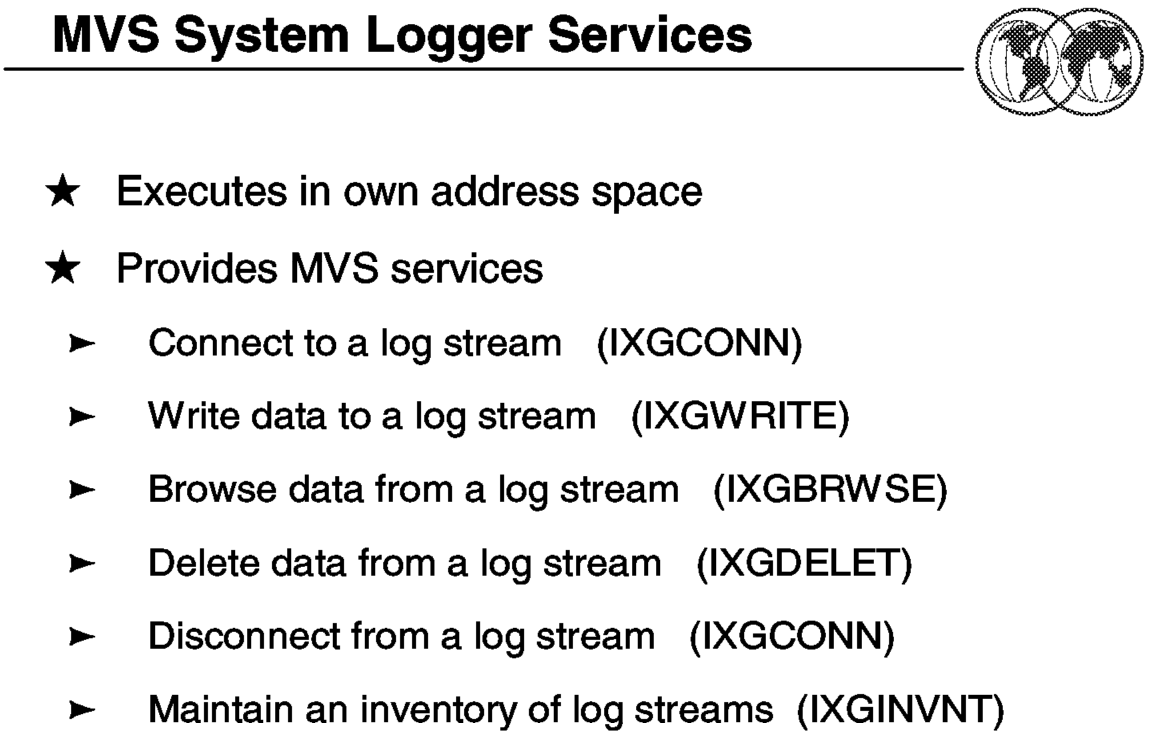


Figure 46. System logger with Coupling Facility

2.1.2 System logger with Coupling Facility

The visual shows how a Coupling Facility log stream spans two levels of storage: the Coupling Facility for interim storage and DASD log data set for more permanent storage. When the Coupling Facility space for the log stream fills, the data is offloaded to DASD log data sets. A Coupling Facility log stream can contain data from multiple systems, allowing a system logger application to merge data from systems across the sysplex.

When a system logger application writes a log block to a Coupling Facility log stream, system logger writes it first to a Coupling Facility list structure. System logger requires that a Coupling Facility list structure be associated with each log stream. When the Coupling Facility structure space allocated for the log stream reaches the installation-defined threshold, system logger moves (offloads) the log blocks from the Coupling Facility structure to VSAM linear DASD data sets, so that the Coupling Facility space for the log stream can be used to hold new log blocks. From a user′s point of view, the actual location of the log data in the log stream is transparent.



MVS System Logger Services

- ★ Executes in own address space
- ★ Provides MVS services
  - ➤ Connect to a log stream (IXGCONN)
  - ➤ Write data to a log stream (IXGWRITE)
  - ➤ Browse data from a log stream (IXGBRWSE)
  - ➤ Delete data from a log stream (IXGDELET)
  - ➤ Disconnect from a log stream (IXGCONN)
  - ➤ Maintain an inventory of log streams (IXGINVNT)

Figure 47. MVS system logger services

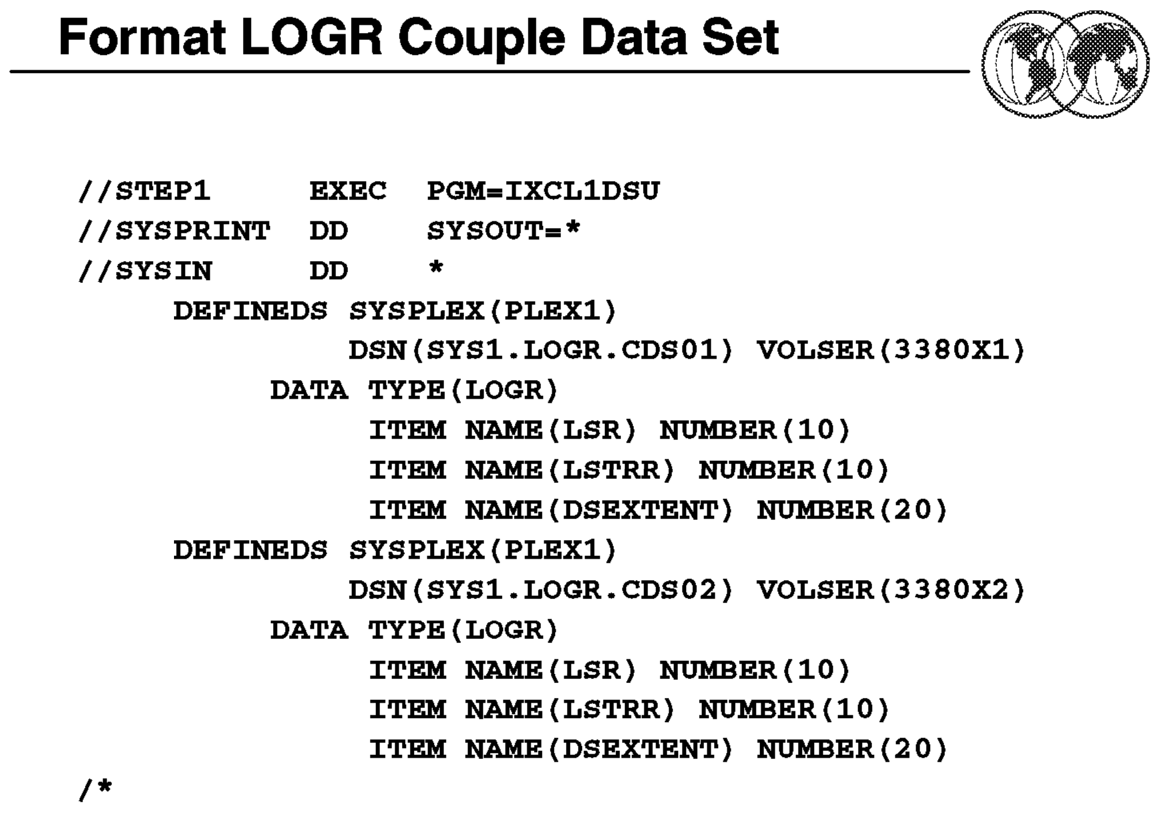
2.1.3 MVS system logger services

The system logger component resides in its own address space on each system in a sysplex. The system logger component does the following:

- Provides a set of system services that allow an application to:
  - Connect to and disconnect from a log stream (IXGCONN)
  - Browse a log stream (IXGBRWSE)
  - Write to a log stream (IXGWRITE)
  - Delete data from a log stream (IXGDELET)
  - Define, update, and delete log stream and Coupling Facility list structure definitions in the LOGR policy, (IXGINVNT service or IXCMIAPU service)

The following services contain parameters for both authorized and unauthorized programs: IXGCONN, IXGBRWSE, IXGWRITE, and IXGDELET. All other system logger services and their parameters can be used by any program.

We recommend that installations use Security Authorization Facility (SAF) to control access to system logger resources such as log streams or Coupling Facility structures associated with log streams.



Format LOGR Couple Data Set

```
//STEP1     EXEC  PGM=IXCL1DSU
//SYSPRINT  DD    SYSOUT=*
//SYSIN     DD    *
     DEFINEDS SYSPLEX(PLEX1)
              DSN(SYS1.LOGR.CDS01) VOLSER(3380X1)
          DATA TYPE(LOGR)
               ITEM NAME(LSR) NUMBER(10)
               ITEM NAME(LSTRR) NUMBER(10)
               ITEM NAME(DSEXTENT) NUMBER(20)
     DEFINEDS SYSPLEX(PLEX1)
              DSN(SYS1.LOGR.CDS02) VOLSER(3380X2)
          DATA TYPE(LOGR)
               ITEM NAME(LSR) NUMBER(10)
               ITEM NAME(LSTRR) NUMBER(10)
               ITEM NAME(DSEXTENT) NUMBER(20)
/*
```

Figure 48. Format LOGR couple data set

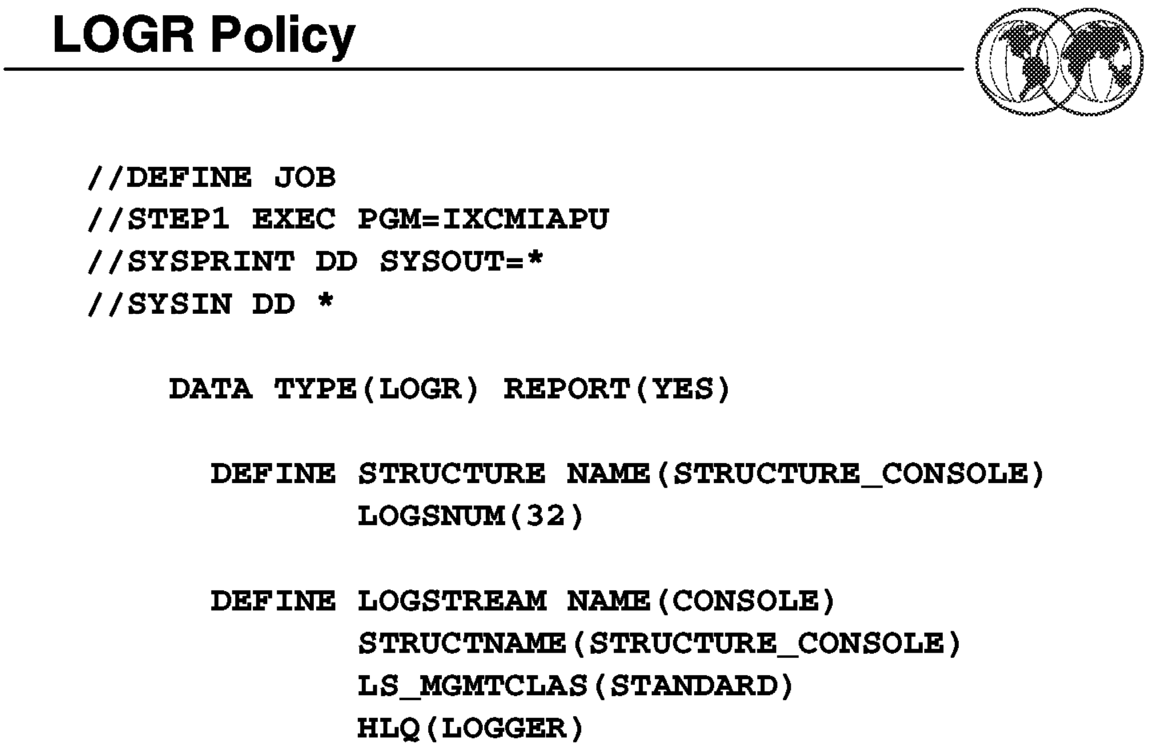
2.1.4 Format LOGR couple data set

The visual shows sample JCL to run the format utility for formatting the LOGR couple data sets for the system logger service. The sample shows formatting for both a primary and alternate LOGR couple data set for the LOGR policy.

2.1.4.1 DASD log data sets

By default, each log stream is limited to a maximum of 168 log data sets. You can, if necessary, increase the number of log data sets available for log streams using the DSEXTENT parameter in an OS/390 Release 3 or higher LOGR couple data set. If you have log streams that will exceed 168 DASD log data sets, perhaps because you retain data in your log stream for a long period of time, increase the number of directory extents available for the sysplex on the DSEXTENT parameter in an OS/390 Release 3 or higher LOGR couple data set using the format utility. Each additional directory extent specified goes into a common pool available to any log stream in the sysplex.

System logger allocates these directory extents as needed. Each directory extent allows a log stream to extend beyond 168 log data sets. Whenever all the log data sets in a data set directory extent have been physically deleted, system logger returns the directory extent record to the available pool of directory extents for the sysplex. Each log stream has one data set directory extent which is part of the log stream record, not part of the data set directory extent pool. This permanent directory extent is always used first for a log stream, before retrieving a directory extent from the common data set directory pool. This permanent extent does not revert to the common pool when unused and is not available for use by any other log stream, it is held for the owning log stream.



LOGR Policy

```
//DEFINE JOB
//STEP1 EXEC PGM=IXCMIAPU
//SYSPRINT DD SYSOUT=*
//SYSIN DD *

    DATA TYPE(LOGR) REPORT(YES)

      DEFINE STRUCTURE NAME(STRUCTURE_CONSOLE)
             LOGSNUM(32)

      DEFINE LOGSTREAM NAME(CONSOLE)
             STRUCTNAME(STRUCTURE_CONSOLE)
             LS_MGMTCLAS(STANDARD)
             HLQ(LOGGER)
```

Figure 49. LOGR policy

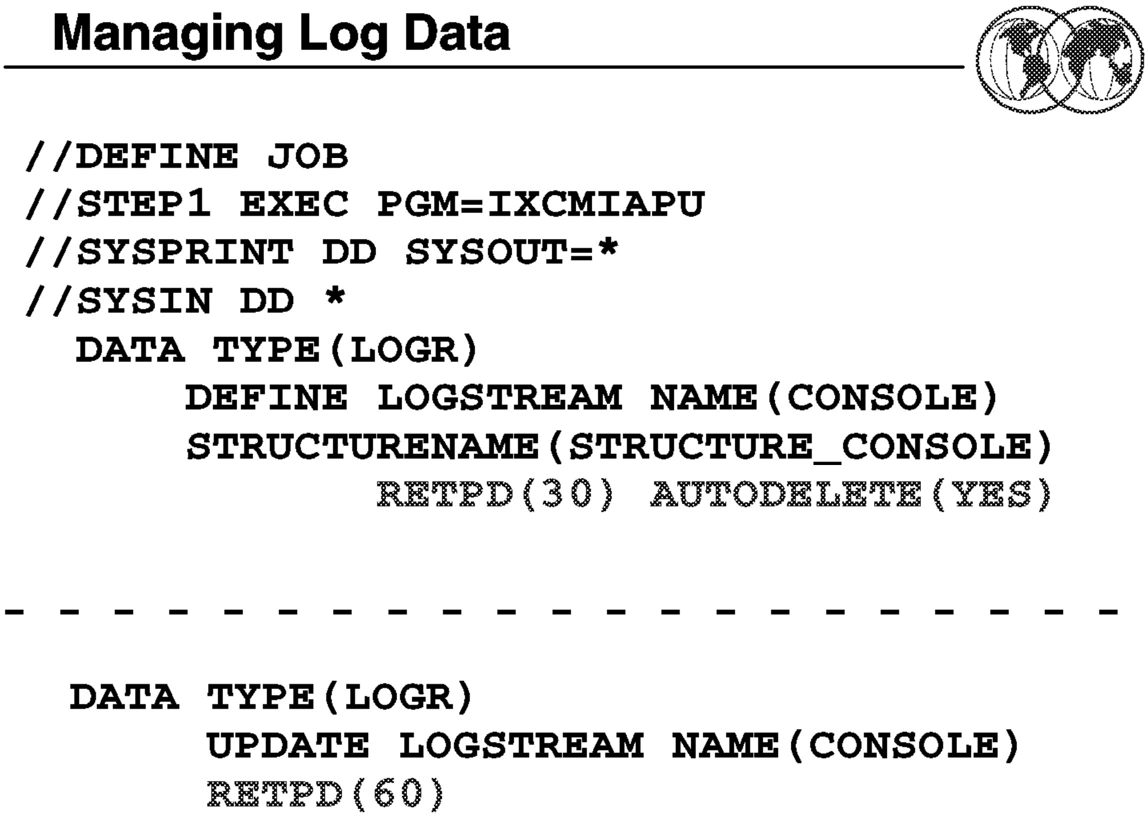
2.1.5 LOGR policy

The visual shows sample JCL to run the utility to define list entries in the LOGR policy. An example of how to use IXCMIAPU with system logger is shown in member IFBLSJCL in A.1, "LOGR IXCMIAPU sample member" on page 347. This is the member that is shipped in SYS1.SAMPLIB.

You can only have one LOGR policy for the sysplex, even if you have primary and alternate LOGR couple data sets. System logger will only use the policy information contained in the primary LOGR couple data set. Because there is only one LOGR policy allowed for a sysplex, you cannot specify the DEFINE POLICY NAME parameter in the LOGR policy, nor can you issue the `SETXCF START,POLICY` command to activate a policy. Updates made to the LOGR policy are made in the primary LOGR couple data set. If you switch to your alternate LOGR couple data set, system logger copies the LOGR policy from the old primary couple data set into the new one.

The LOGR policy includes the following:

- Log stream definitions
- Coupling facility list structure definitions, if applicable
- Data containing the current state of a log stream (for example, whether a log stream is currently connected to the Coupling Facility structure)



Managing Log Data

```
//DEFINE JOB
//STEP1 EXEC PGM=IXCMIAPU
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
  DATA TYPE(LOGR)
       DEFINE LOGSTREAM NAME(CONSOLE)
       STRUCTURENAME(STRUCTURE_CONSOLE)
                RETPD(30) AUTODELETE(YES)
```

```
  DATA TYPE(LOGR)
       UPDATE LOGSTREAM NAME(CONSOLE)
       RETPD(60)
```

Figure 50. Managing log data

2.1.6 Managing log data

For installations with an active primary LOGR couple data set at an OS/390 Release 3 level or higher, system logger provides support to make it easier to archive log data and manage the amount of data kept in a log stream. This applies to both Coupling Facility and DASD-only log streams. You can define a retention period and automatic deletion policy for each log stream. The retention period and automatic deletion policy are specified on the RETPD and AUTODELETE parameters in an OS/390 Release 3 higher LOGR couple data set to help manage:

- How much data you keep in a log stream
- How long you keep data in the log stream

The RETPD and AUTODELETE parameters are used together to set up a retention period and automatic deletion for a log stream. You can specify the RETPD and AUTODELETE parameters in an OS/390 Release 3 level or higher LOGR couple data set using either the IXCMIAPU utility or the IXGINVNT service in a program.

2.1.6.1 REPTD parameter

The RETPD parameter allows you to specify a retention period for a log stream. On RETPD, you specify the number of days that you want to keep data in the log stream, even if the data has been marked for deletion using IXGDELET. For example, if you specify RETPD(7) in the LOGR policy for a log stream, the retention period for data in that log stream is seven days from the time the data is written to the log stream by the application. System logger processes the retention period on a log data set basis. Once the retention period for the entire log data set has expired, the data set is eligible for deletion. System logger may not physically delete data as soon as it hits the retention period. The point at which system logger physically deletes the data depends on when the data set fills and what you specify on the AUTODELETE parameter.

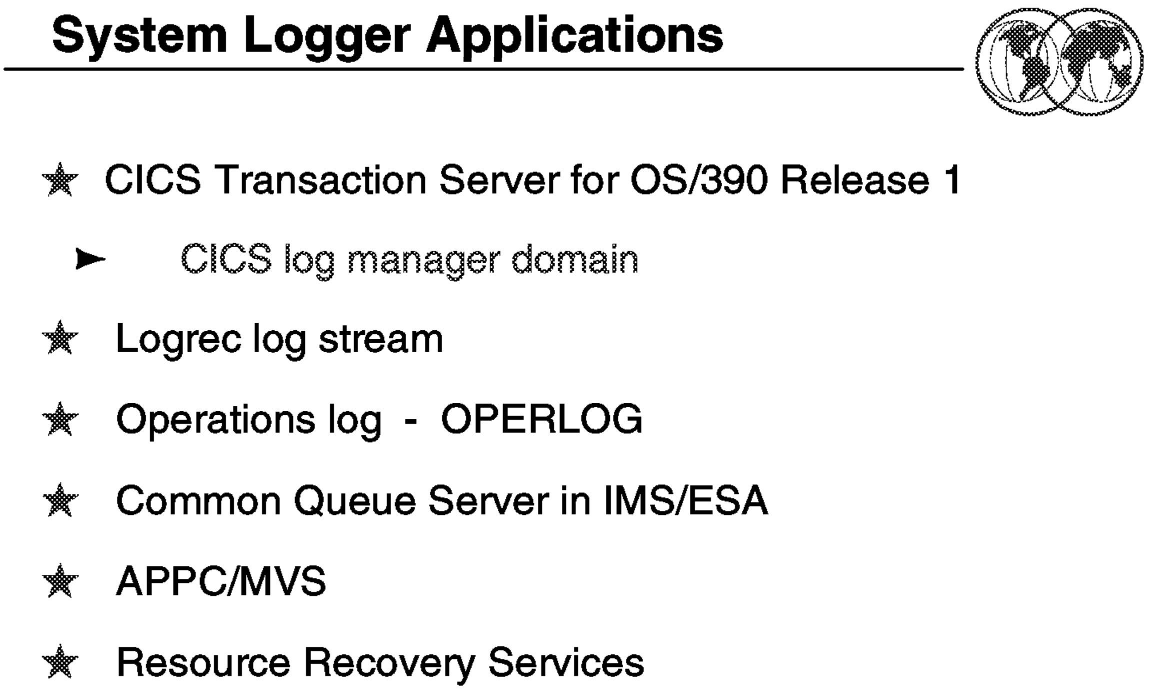
2.1.6.2 AUTODELETE parameter

Use AUTODELETE(NO) to indicate that you do not want an automatic deletion policy for a log stream. AUTODELETE(NO) means that the log data can be physically deleted only after the log data has been marked for deletion via IXGDELET and after any retention period specified for the log stream has expired. AUTODELETE(NO) is the default.

Use AUTODELETE(YES) to indicate that you want an automatic deletion policy to limit how long data is kept in the log stream. AUTODELETE(YES) means that log data can be physically deleted either when the data is marked for deletion or when a retention period specified for the log stream expires. Use care when specifying AUTODELETE(YES) because automatic deletion is designed to speed physical deletion of log data, which can mean deletion of data that an application needs.

2.1.6.3 UPDATE parameter

The UPDATE LOGSTREAM specification requests that an entry for a coupling facility or DASD-only log stream be updated in the LOGR policy. Except for the RETPD, AUTODELETE, and DIAG parameters, you cannot update a log stream while there are active connections (active or failed) to it. Using the IXCMAIPU utility, as shown on the visual, you can UPDATE the RETPD parameter.



System Logger Applications

- CICS Transaction Server for OS/390 Release 1
  - CICS log manager domain
- Logrec log stream
- Operations log - OPERLOG
- Common Queue Server in IMS/ESA
- APPC/MVS
- Resource Recovery Services

Figure 51. System logger applications

2.1.7 System logger applications

System logger applications can be supplied by IBM, independent software vendors, or written at your installation. For each of these applications some information is necessary for set-up and use. Some of the IBM-supplied system logger applications are:

- CICS log manager

  CICS log manager is a CICS system logger application that replaces the journal control management function. It provides a focal point for all CICS system log, forward recovery log, and user journal output within a sysplex and flexible mapping of CICS journals onto log streams. CICS log manager also enables faster CICS restart, dual logging, and flexible log and journal archiving.

- Logrec log stream

  Logrec log stream is an MVS system logger application that records hardware failures, selected software errors, and selected system conditions across the sysplex. Using a logrec log stream rather than a logrec data set for each system can streamline logrec error recording.

- Operations log (OPERLOG)

  Operations log is an MVS system logger application that records and merges messages about programs and system functions (the hardcopy message set) from each system in a sysplex that activates OPERLOG. Use OPERLOG rather than the system log (SYSLOG) as your hardcopy medium when you need a permanent log about operating conditions and maintenance for all systems in a sysplex.

- IMS common shared queues (CQS) log manager

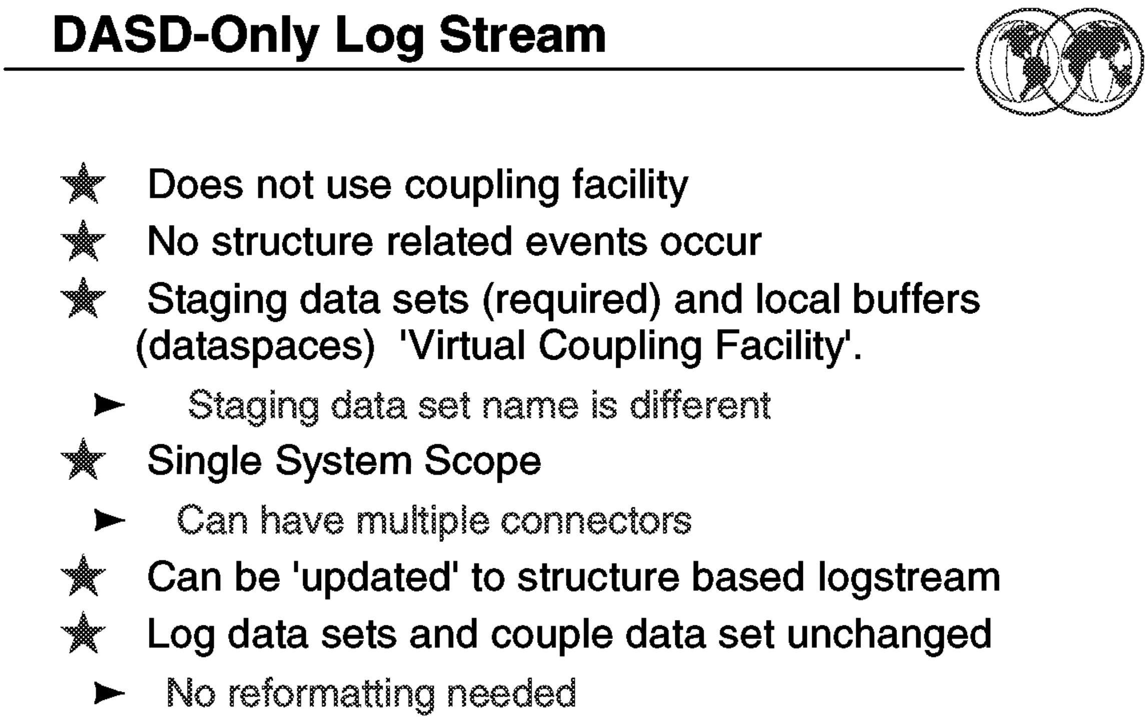
  IMS common shared queues (CQS) log manager is a system logger application that records the information necessary for CQS to recover structures and restart. CQS writes log records into a separate log stream for each Coupling Facility list structure pair that it uses. IMS CQS log manager uses Coupling Facility-based log streams.

- APPC/MVS

  APPC/MVS is an MVS system logger application that records events related to protected conversations. An installation-defined log stream is required for APPC/MVS to protect conversations, through its participation in resource recovery.

- Resource recovery services (RRS)

  RRS is an MVS system logger application that records events related to protected resources. RRS records these events in five log streams that are shared by systems in a sysplex. To set up and use the RRS system logger application refer to “Preparing to Use System Logger Applications” in *OS/390 MVS Programming: Resource Recovery*, GC28-1739.



DASD-Only Log Stream

- Does not use coupling facility
- No structure related events occur
- Staging data sets (required) and local buffers (dataspaces) 'Virtual Coupling Facility'.
  - Staging data set name is different
- Single System Scope
  - Can have multiple connectors
- Can be 'updated' to structure based logstream
- Log data sets and couple data set unchanged
  - No reformatting needed

Figure 52. DASD-only log streams

2.2 DASD-only log streams

The system logger function has been enhanced, beginning with OS/390 Release 4, to include support for DASD-only log streams. DASD-only log streams do not use Coupling Facility storage. The Coupling Facility is used by system logger as the interim storage for log stream data before it is offloaded (hardened) to DASD log data sets. For a DASD-only log stream, however, the log data is stored in local storage buffers before being offloaded (hardened) to DASD log data sets.

A DASD-only log stream has a single-system scope; only one system at a time can connect to a DASD-only log stream. Multiple applications from the same system can, however, simultaneously connect to a DASD-only log stream. When a system logger application writes a log block to a DASD-only log stream, system logger writes it first to the local storage buffers for the system and then automatically duplexes it to a DASD staging data set associated with the log stream.

A DASD-only log stream can be upgraded to a Coupling Facility log stream by updating the log stream definition in the LOGR policy couple data set to associate a Coupling Facility structure with the log stream.

DASD-only Logger

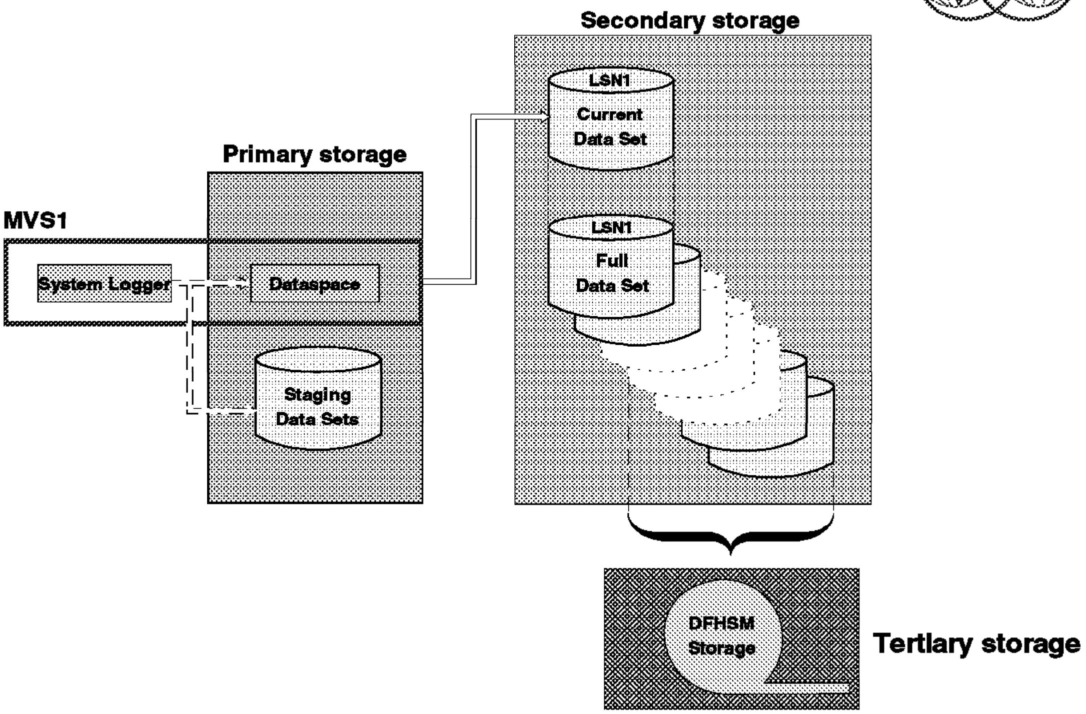


Figure 53. DASD-only logger

2.2.1 DASD-only logger

The visual shows a DASD-only log stream spanning two levels of storage: local storage buffers for interim storage, which is then off-loaded to DASD log data sets for more permanent storage.

A DASD-only log stream has a single-system scope; only one system at a time can connect to a DASD-only log stream. Multiple applications from the same system can, however, simultaneously connect to a DASD-only log stream.

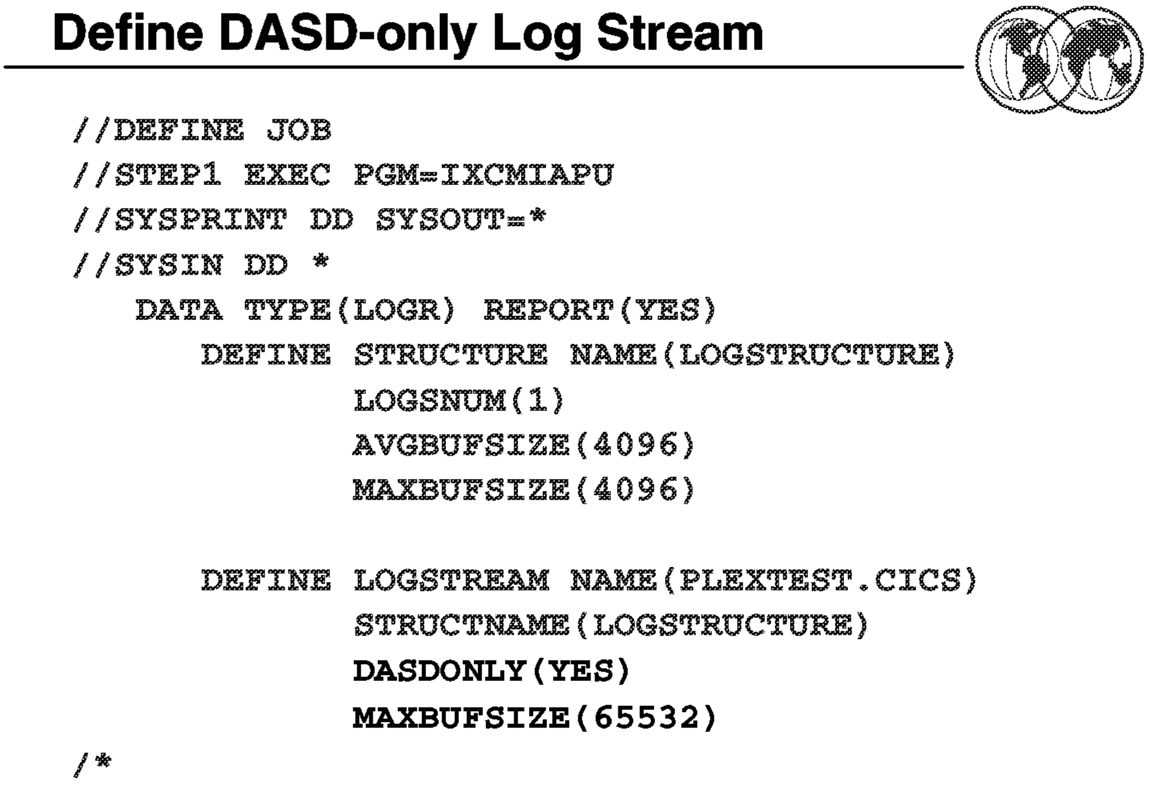
When a system logger application writes a log block to a DASD-only log stream, system logger writes it first to the local storage buffers for the system and duplexes it to a DASD staging data set associated with the log stream. When the staging data set space allocated for the log stream reaches the installation-defined threshold, system logger off-loads the log blocks from local storage buffers to VSAM linear DASD data sets. From a user′s point of view, the actual location of the log data in the log stream is transparent.

Both a DASD-only log stream and a Coupling Facility log stream can have data in multiple DASD log data sets; as a log stream fills log data sets on DASD, system logger automatically allocates new ones for the log stream.

For a DASD-only log stream, system logger automatically duplexes log data from the system′s local storage buffers to DASD staging data sets. System logger uses VSAM linear DASD data sets for staging data sets.

Note: The staging data set naming convention is different for DASD-only and Coupling Facility log streams.

For DASD-only log streams, staging data sets are a required part of the system logger configuration. System logger automatically duplexes data to the staging data set for the system at the same time it writes the data local storage buffers. We recommend that you plan for a staging data set for each system, using SMS to manage them. If you do not plan for the staging data sets using SMS, system logger will automatically allocate staging data sets for a system connecting to a DASD-only log stream, using the defaults outlined in the next visual.



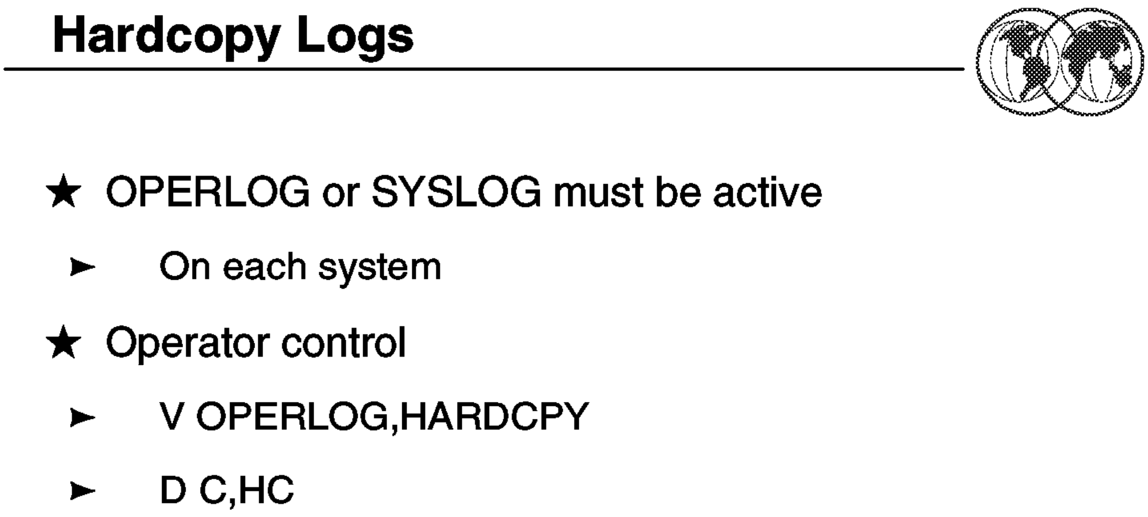
Define DASD-only Log Stream

```
//DEFINE JOB
//STEP1 EXEC PGM=IXCMIAPU
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
   DATA TYPE(LOGR) REPORT(YES)
      DEFINE STRUCTURE NAME(LOGSTRUCTURE)
             LOGSNUM(1)
             AVGBUFSIZE(4096)
             MAXBUFSIZE(4096)

      DEFINE LOGSTREAM NAME(PLEXTEST.CICS)
             STRUCTNAME(LOGSTRUCTURE)
             DASDONLY(YES)
             MAXBUFSIZE(65532)
/*
```

Figure 54. Define DASD-only log stream

2.2.2 Define DASD-only log stream

To define a log stream as DASD-only, specify DASDONLY(YES) on the definition for a log stream that is not associated with a coupling facility structure. When you define a DASD-only log stream, you can also specify MAXBUFSIZE on the DEFINE LOGSTREAM request to define the maximum buffer size that can be written to the DASD-only log stream.



Hardcopy Logs

★ OPERLOG or SYSLOG must be active

➤ On each system

★ Operator control

➤ V OPERLOG,HARDCPY

➤ D C,HC

Figure 55. Hardcopy logs

2.3 Hardcopy logs

The term *hardcopy log* can refer to:

- The system log (SYSLOG)
- The operations log (OPERLOG)
- The device used to print hardcopy messages
- The data set containing hardcopy messages
- The actual printed copy of the hardcopy messages

Hardcopy processing is required in a sysplex.

Hardcopy processing allows your installation to have a permanent record of system activity and helps you audit the use of operator commands. The group of messages and commands that are recorded is called the hardcopy message set. The system log, operations log, or MCS printer that receives messages is called the hardcopy medium. You can specify a group of console devices that can serve as backup devices for the hardcopy medium. You can also allow an extended MCS console to receive hardcopy messages from one or more systems in a sysplex.

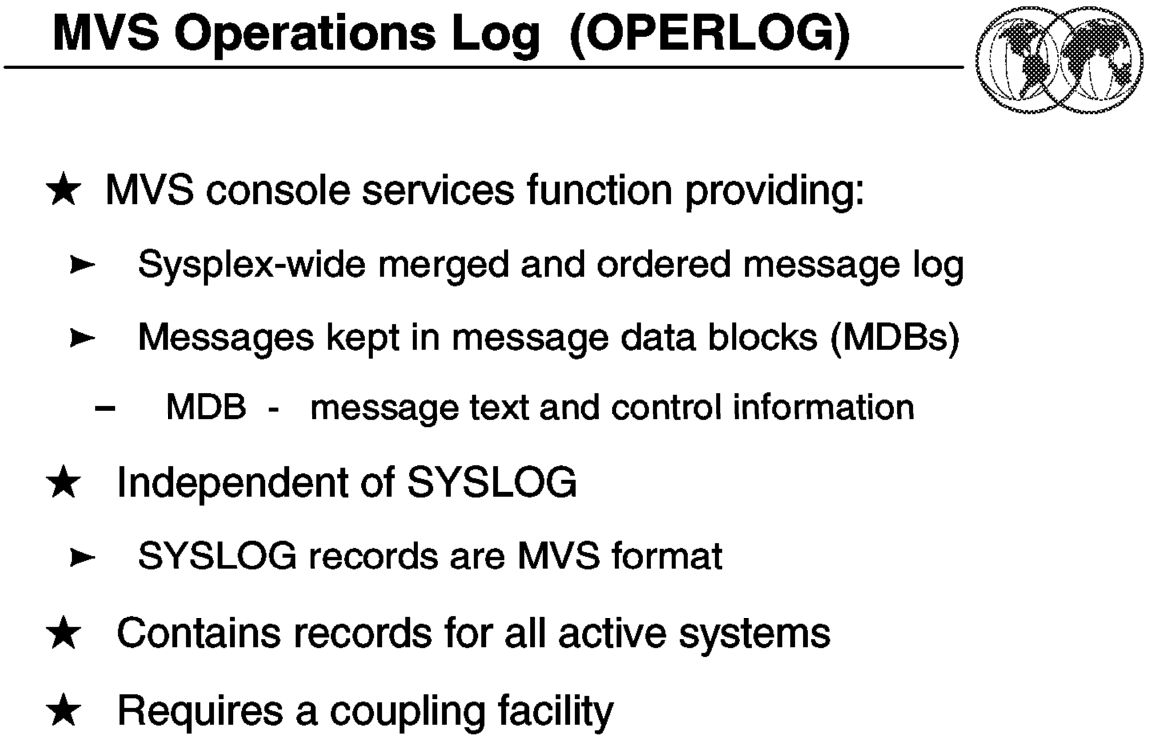
In a JES2 system or a JES3 system running on Release 5.2.1 or higher, the hardcopy message set is usually sent to the current active log, either the system log or the operations log, or both, but may be sent to a printer console, if the installation chooses. The hardcopy message set is defined at system initialization, in the CONSOLxx parmlib member, and may be subsequently changed by the VARY command.

The OPERLOG status on each system can be displayed with the D C,HC operator command. The VARY OPERLOG,HARDCPY command activates the OPERLOG and the VARY OPERLOG,HARDCPY,OFF command deactivates the OPERLOG on a system.

The system log (SYSLOG) is a direct access data set that stores messages and commands. It resides in the primary job entry subsystem′s spool space. It can be used by application and system programmers (through the WTL macro) to record communications about programs and system functions. You can use the LOG command to add an entry to the system log.

Several kinds of information can appear in the system log:

- Job time, step time, and data from the JOB and EXEC statements of completed jobs entered by user-written routines
- Operating data entered by programs using a write to log (WTL) macro instruction
- Descriptions of unusual events that you enter using the LOG command
- The hardcopy message set



MVS Operations Log (OPERLOG)

★ MVS console services function providing:

- ➤ Sysplex-wide merged and ordered message log
- ➤ Messages kept in message data blocks (MDBs)
  - – MDB - message text and control information

★ Independent of SYSLOG

- ➤ SYSLOG records are MVS format

★ Contains records for all active systems

★ Requires a coupling facility

Figure 56. MVS operations log (OPERLOG)

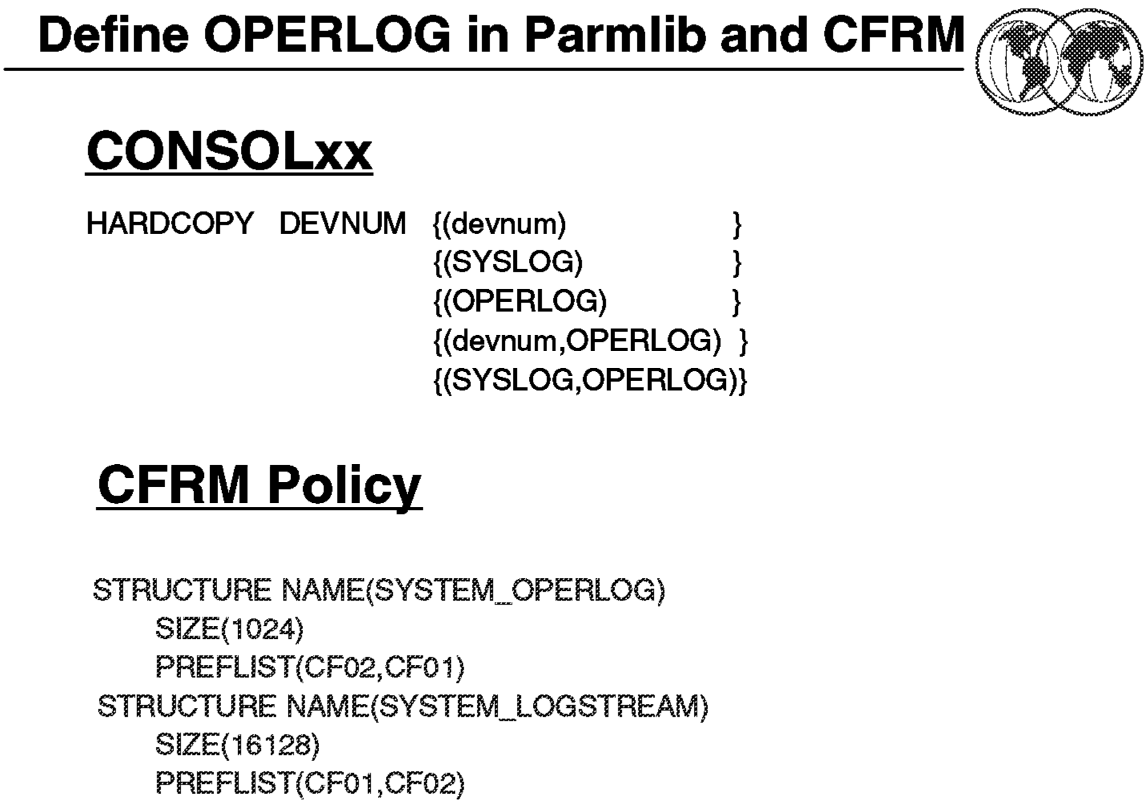
2.4 MVS operations log (OPERLOG)

The OPERLOG provides a sysplex-wide merged and chronologically ordered message log.

- The messages are logged using message data blocks (MDB), which provide more data than is recorded in the SYSLOG. You can use member IEAMDBLG, in SYS1.SAMPLIB, to convert OPERLOG records into SYSLOG format.
- The OPERLOG is maintained by the MVS system logger.
- Only the systems in the sysplex that have specified and activated the operations log will have their records sent to OPERLOG.

The operations log (OPERLOG) is a log stream that uses the system logger to record and merge communications about programs and system functions from each system in a sysplex. Only the systems in a sysplex that have specified and activated the operations log will have their records sent to OPERLOG. For example, if a sysplex has three systems, SYSA, SYSB, and SYSC, but only SYSA and SYSB activate the operations log, then only SYSA and SYSB will have their information recorded in the operations log.

You can also use OPERLOG as a DASD-only log stream. This method is only suitable for a single system sysplex, because a DASD-only log stream is single-sysplex in scope and you can only have one OPERLOG log stream per sysplex. This means that if you make OPERLOG a DASD-only log stream, only one system can access it.



Define OPERLOG in Parmlib and CFRM

CONSOLxx

```
HARDCOPY  DEVNUM  {(devnum)            }
                  {(SYSLOG)            }
                  {(OPERLOG)           }
                  {(devnum,OPERLOG)    }
                  {(SYSLOG,OPERLOG)}
```

CFRM Policy

```
STRUCTURE NAME(SYSTEM_OPERLOG)
     SIZE(1024)
     PREFLIST(CF02,CF01)
STRUCTURE NAME(SYSTEM_LOGSTREAM)
     SIZE(16128)
     PREFLIST(CF01,CF02)
```

Figure 57. Define OPERLOG in parmlib and CFRM

2.4.1 Define OPERLOG in parmlib and CFRM

HARDCOPY identifies the hardcopy medium and defines the hardcopy message set.

Where:

DEVNUM Specifies the output device.

devnum Specifies the device number of a printer console that is to be the hardcopy medium.

SYSLOG Indicates that the system log is to be the hardcopy medium.

OPERLOG Indicates that the operations log will be activated and will receive the hardcopy message set. You can specify OPERLOG alone or with devnum or SYSLOG. When you specify OPERLOG with devnum or SYSLOG, both OPERLOG and devnum or SYSLOG will receive the hardcopy message set.

If you do not specify devnum, SYSLOG, or OPERLOG, the system defaults to the hardcopy medium (devnum or SYSLOG) if it is active; otherwise, the system rejects the command. The system will not deactivate the operations log unless OPERLOG is specified.

Before you can begin using the operations log, you must define a log stream using the system logger services. You should define a logstream with name *SYSPLEX.OPERLOG*. This you do in either the data administrative utility or in the IXGINVNT macro.

See *OS/390 MVS Programming: Assembler Services Guide*, GC28-1762, for information about using system logger services.

In your CFRM policy, you must provide the definitions for system logger structures. The CFRM policy defines the structures: the amount of Coupling Facility storage to be used for each structure, an ordered preference list of Coupling Facilities in which each structure should reside, and an unordered exclusion list of structure names that should not be allocated in the same Coupling Facility as the specified structure.

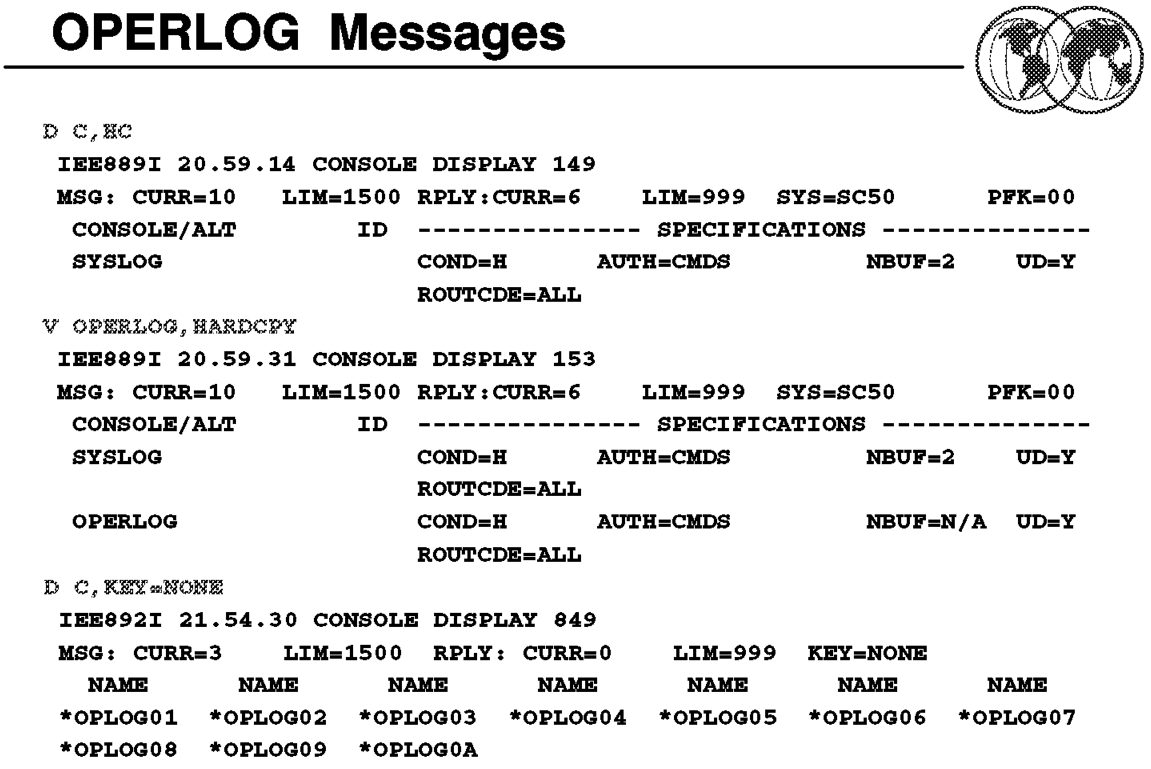
2.4.1.1 LOGR policy

The system logger component manages log streams based on the policy information that installations place in the LOGR couple data set. The LOGR couple data set must:

- Be accessible by all the systems in the sysplex
- Be formatted using the IXCL1DSU utility
- Contain policy information, which is defined using the IXCMIAPU utility or the IXGINVNT service

You use the IXCMIAPU utility to create or update the LOGR administrative data. The following example shows definitions for a dedicated OPERLOG structure and a shared log stream structure:

```
//DEFCFSTR  JOB (999,POK),′ANDILE′,CLASS=A,REGION=4M,
//            MSGCLASS=X,TIME=10,MSGLEVEL=(1,1),NOTIFY=&SYSUID
//DEFINE   EXEC PGM=IXCMIAPU
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
 DATA TYPE (LOGR)
    DEFINE STRUCTURE NAME(SYSTEM_OPERLOG)
           LOGSNUM(32)
           AVGBUFSIZE(4096)
           MAXBUFSIZE(65532)
    DEFINE LOGSTREAM NAME(SYSPLEX.OPERLOG)
           STG_DUPLEX(NO) LS_DATACLAS(SHARE33)
           LS_SIZE(2048)
           LOWOFFLOAD(0) HIGHOFFLOAD(80)
           STRUCTNAME(SYSTEM_OPERLOG) HLQ(IXGLOGR)
/*
```



OPERLOG Messages

```
D C,HC
 IEE889I 20.59.14 CONSOLE DISPLAY 149
 MSG: CURR=10   LIM=1500 RPLY:CURR=6    LIM=999  SYS=SC50      PFK=00
  CONSOLE/ALT        ID  --------------- SPECIFICATIONS --------------
  SYSLOG                 COND=H      AUTH=CMDS         NBUF=2    UD=Y
                         ROUTCDE=ALL
V OPERLOG,HARDCPY
 IEE889I 20.59.31 CONSOLE DISPLAY 153
 MSG: CURR=10   LIM=1500 RPLY:CURR=6    LIM=999  SYS=SC50      PFK=00
  CONSOLE/ALT        ID  --------------- SPECIFICATIONS --------------
  SYSLOG                 COND=H      AUTH=CMDS         NBUF=2    UD=Y
                         ROUTCDE=ALL
  OPERLOG                COND=H      AUTH=CMDS         NBUF=N/A  UD=Y
                         ROUTCDE=ALL
D C,KEY=NONE
 IEE892I 21.54.30 CONSOLE DISPLAY 849
 MSG: CURR=3    LIM=1500  RPLY: CURR=0    LIM=999  KEY=NONE
   NAME       NAME       NAME       NAME       NAME       NAME       NAME
 *OPLOG01   *OPLOG02   *OPLOG03   *OPLOG04   *OPLOG05   *OPLOG06   *OPLOG07
 *OPLOG08   *OPLOG09   *OPLOG0A
```

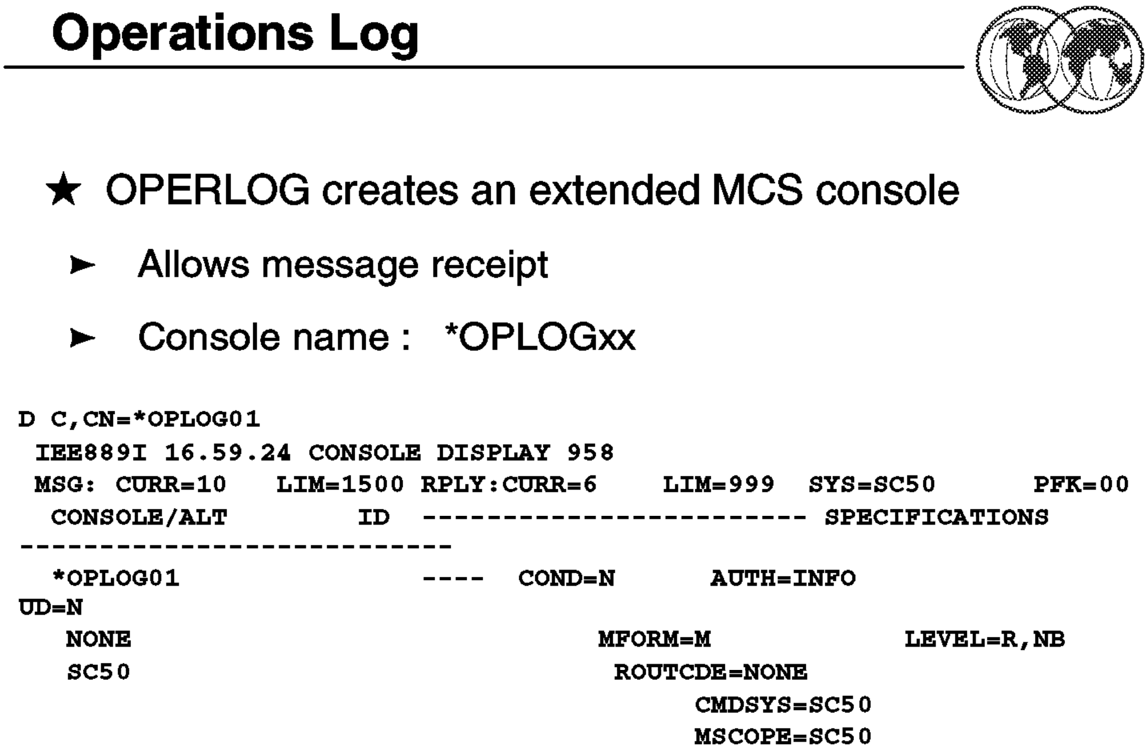
Figure 58. OPERLOG messages

2.4.2 OPERLOG messages

If OPERLOG is not defined in parmlib, the operator can activate the OPERLOG with the following command, providing the MVS system logger is activated:

```
V OPERLOG,HARDCPY
IEE889I 20.59.31 CONSOLE DISPLAY 153
MSG: CURR=10   LIM=1500 RPLY:CURR=6    LIM=999  SYS=SC50      PFK=00
 CONSOLE/ALT        ID  --------------- SPECIFICATIONS --------------
 SYSLOG                 COND=H      AUTH=CMDS         NBUF=2    UD=Y
                        ROUTCDE=ALL
 OPERLOG                COND=H      AUTH=CMDS         NBUF=N/A  UD=Y
                        ROUTCDE=ALL
```

The OPERLOG function internally activates an extended MCS console to receive messages for transcription to the log. Each system that is using the OPERLOG has an active console whose name is *OPLOGxx, where xx is the XCF system slot number of that system, as shown in the visual.



Operations Log

- ★ OPERLOG creates an extended MCS console
  - ➤ Allows message receipt
  - ➤ Console name :   *OPLOGxx

```
D C,CN=*OPLOG01
 IEE889I 16.59.24 CONSOLE DISPLAY 958
 MSG: CURR=10   LIM=1500 RPLY:CURR=6    LIM=999  SYS=SC50      PFK=00
  CONSOLE/ALT        ID  ------------------------ SPECIFICATIONS
---------------------------
  *OPLOG01               ----  COND=N      AUTH=INFO
UD=N
   NONE                             MFORM=M            LEVEL=R,NB
   SC50                              ROUTCDE=NONE
                                          CMDSYS=SC50
                                          MSCOPE=SC50
```

Figure 59. Operations log

2.5 Operations log

The OPERLOG function internally activates an extended MCS console to receive messages for transcription to the log. Each system that is using the OPERLOG has an active console whose name is *OPLOGxx, where xx is the XCF system slot number of that system, as shown in the visual.

This allows the messages to be received by the EMCS console.

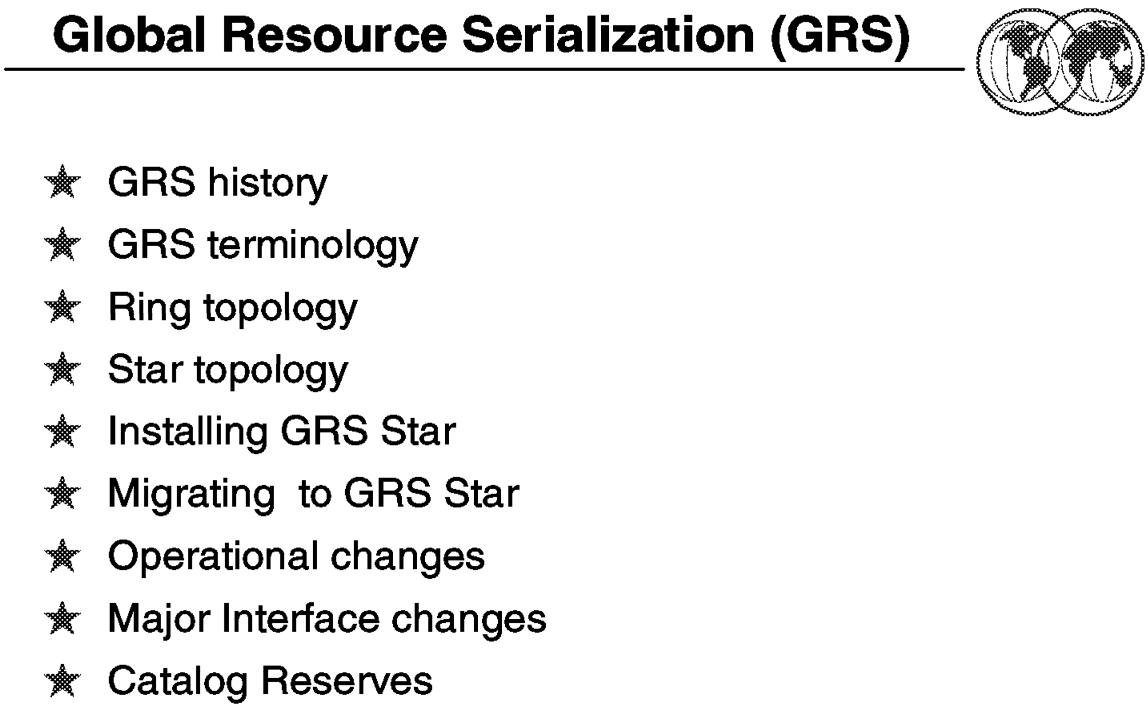
Chapter 3. Global resource serialization

In a multitasking, multiprocessing environment, resource serialization is the technique used to coordinate access to resources that are used by more than one program. When multiple users share data, a way to control access to that data is needed. Users that update data, for example, need exclusive access to that data. If several users tried to update the same data at the same time, the result would be a data that could be incorrect or corrupted. In contrast, users who only read data can safely access the same data at the same time.

Global resource serialization (GRS) offers the control needed to ensure the integrity of resources in a multisystem environment. Combining the systems that access shared resources into a global resource serialization complex enables you to serialize resources across multiple systems. In a global resource serialization complex, programs can serialize access to data sets on shared DASD volumes at the data set level rather than at the DASD volume level. A program on one system can access one data set on a shared volume while other programs on any system can access other data sets on the volume. Because GRS enables jobs to serialize resources at the data set level it can reduce contention for these resources and minimize the chance of an interlock occurring between systems.

Access to a resource can be requested as *exclusive* or *shared*. When GRS grants shared access to a resource, no exclusive users will be granted access to the resource simultaneously. Likewise, when GRS grants exclusive access to a resource, all other requestors for the resource will wait until the exclusive requestor frees the resource.

If you build a multisystem sysplex, you automatically build a global resource serialization complex with it. As the systems join the sysplex the systems join the global resource serialization complex.



Global Resource Serialization (GRS)

- GRS history
- GRS terminology
- Ring topology
- Star topology
- Installing GRS Star
- Migrating to GRS Star
- Operational changes
- Major Interface changes
- Catalog Reserves

Figure 60. Global resource serialization (GRS)

3.1 Global resource serialization (GRS)

In a multisystem sysplex, MVS provides a platform of services that applications and subsystems can exploit. Multisystem applications reside on more than one system and give a single-system image to the users. This requires serialization services that extend across systems.

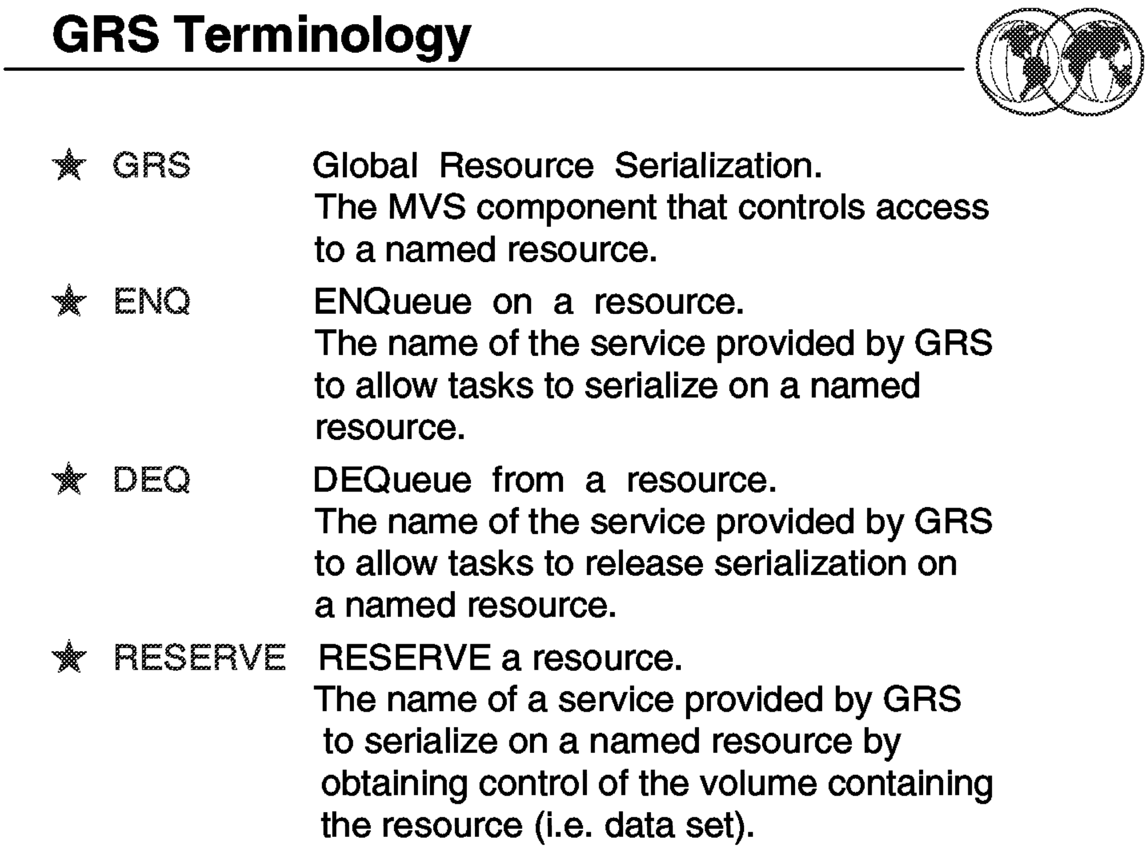
GRS provides the services to ensure the integrity of resources in a multisystem environment. Combining the systems that access shared resources into a global resource serialization complex enables serialization across multiple systems.

GRS has been available in MVS since MVS/SP 1.3. However, activating GRS in cross-system serialization (global) mode in a multisystem complex has been optional. MVS/ESA Version 4 introduced the concept of *guaranteed serialization* in a sysplex, which requires that GRS be active in cross-system serialization mode. Any multisystem sysplex is thus required to activate global serialization for all members of the sysplex.

All previous releases of MVS/ESA and OS/390 that supported GRS consisted of one or more systems connected to each other in a *ring* configuration.

OS/390 Release 2 introduced the ability to design a GRS complex in a *star* configuration. It is still possible to run all OS/390 releases using the ring configuration. The main change in the *star mode* method of processing requests for global resources is that they are managed through a Coupling Facility lock structure, and that each system maintains only its own global requests, while in the *ring* *mode* method of processing, each system in the GRS complex maintains all the requests for all global resources in the complex.

The new star configuration has positive effects on sysplex growth, response time for global requests, and processor and storage resources consumption.



GRS Terminology

★ GRS — Global Resource Serialization.
The MVS component that controls access to a named resource.

★ ENQ — ENQueue on a resource.
The name of the service provided by GRS to allow tasks to serialize on a named resource.

★ DEQ — DEQueue from a resource.
The name of the service provided by GRS to allow tasks to release serialization on a named resource.

★ RESERVE — RESERVE a resource.
The name of a service provided by GRS to serialize on a named resource by obtaining control of the volume containing the resource (i.e. data set).

Figure 61. GRS terminology

3.1.1 Local and global resources

The ENQ service attempts to assign control of a resource to the current requester. The resource to be serialized is identified by name and scope. There are three levels of serialization:

STEP The resource is unique to a single address space.

SYSTEM The resource is unique to a single system.

SYSTEMS The resource is unique to the GRS complex.

The requester also identifies the type of serialization (shared or exclusive).

Note: Even if there is only one system in the GRS complex, the scope SYSTEM and SYSTEMS ENQ requests for the same resource name identify two *different* resources.

The DEQ service releases serialization of one or more held resources.

The ENQ, DEQ, and RESERVE macros identify a resource by its symbolic name. The symbolic name has three parts:

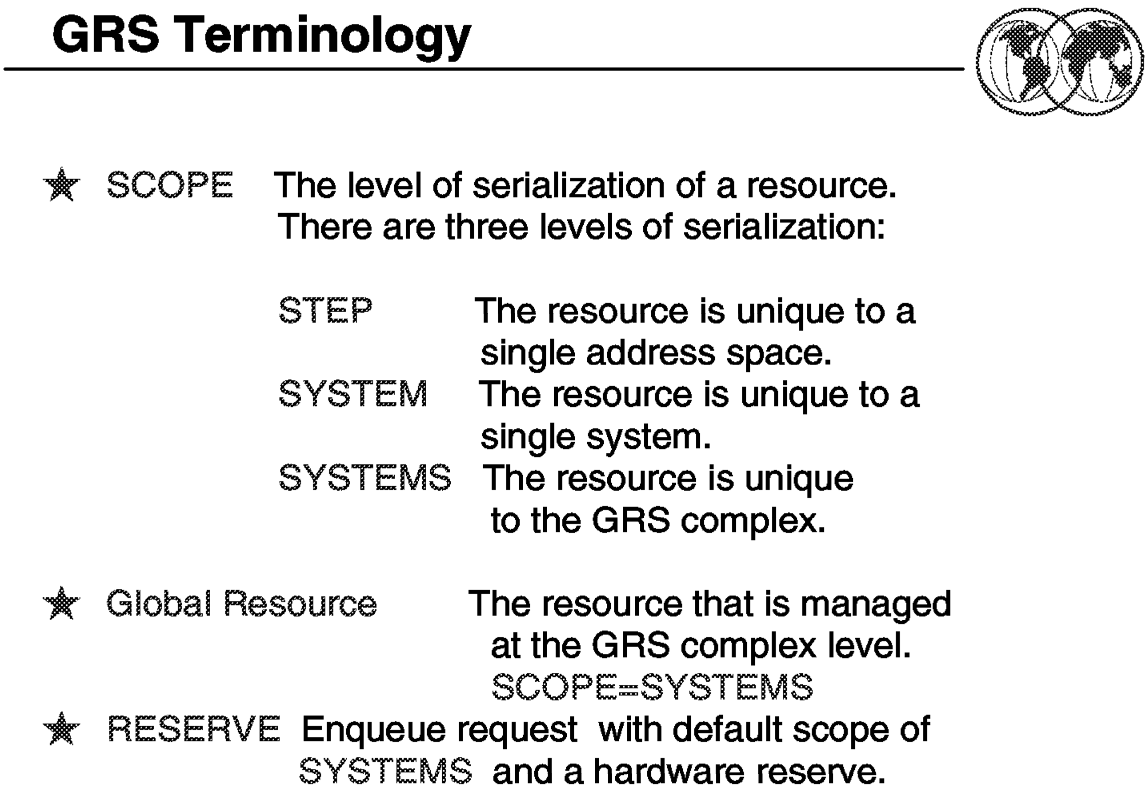
- Major name (QNAME)
- Minor name (RNAME)
- SCOPE (which can be STEP, SYSTEM, or SYSTEMS).

For example, on an ENQ or DEQ macro, a resource might have a symbolic name of APPL01,MASTER,SYSTEM. The major name (qname) is APPL01, the minor name (rname) is MASTER, and the scope is SYSTEM. Global resource serialization identifies each resource by its entire symbolic name. For example, a resource that is specified as A,B,SYSTEMS is considered a different resource from A,B,SYSTEM or A,B,STEP because the scope of each resource different.

When an application uses the ENQ, DEQ, and RESERVE macros to serialize resources, global resource serialization uses resource name lists (RNLs) and the scope on the ENQ, DEQ, or RESERVE macro to determine whether a resource is a local resource or a global resource.

A *local resource* is a resource requested with a scope of STEP or SYSTEM. It is serialized only within the system processing the request for the resource. If a system is not part of a global resource serialization complex, all resources (with the exception of resources serialized with the RESERVE macro), regardless of scope (STEP, SYSTEM, SYSTEMS), are local resources.

A *global resource* is a resource requested with a scope of SYSTEMS. It is serialized among all systems in the complex.



GRS Terminology

★ SCOPE The level of serialization of a resource. There are three levels of serialization:

STEP	The resource is unique to a single address space.
SYSTEM	The resource is unique to a single system.
SYSTEMS	The resource is unique to the GRS complex.

★ Global Resource The resource that is managed at the GRS complex level. SCOPE=SYSTEMS

★ RESERVE Enqueue request with default scope of SYSTEMS and a hardware reserve.

Figure 62. GRS levels of serialization

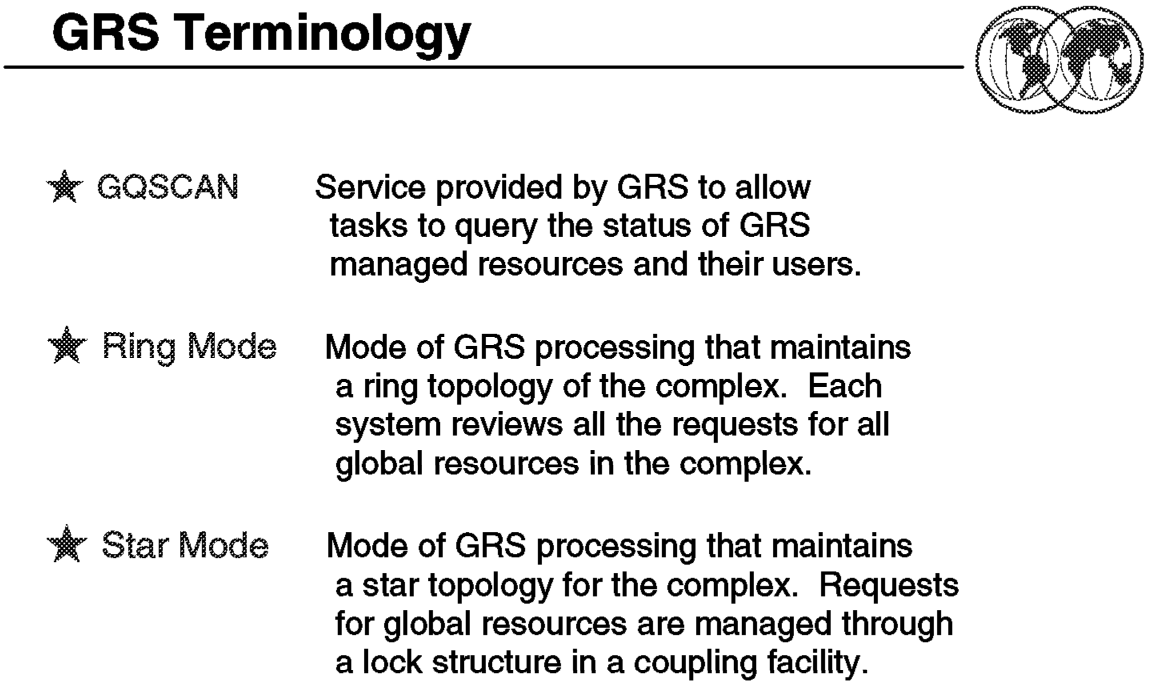
3.1.2 GRS levels of serialization

The RESERVE macro has a scope of SYSTEMS by default, while ENQ and DEQ can have either STEP, SYSTEM, or SYSTEMS.

To ensure that resources are treated as you want them to be without changes to your applications, global resource serialization provides three resource name lists (RNLs):

- The SYSTEM inclusion RNL lists resources requested with a scope of SYSTEM that you want global resource serialization to treat as global resources.
- The SYSTEMS exclusion RNL lists resources requested with a scope of SYSTEMS that you want global resource serialization to treat as local resources.
- The RESERVE conversion RNL lists resources requested on RESERVE macro for which you want global resource serialization to suppress the reserve.

By placing the name of a resource in the appropriate RNL, you can cause global resource serialization to process it as you want. The RNLs enable you to build a global resource serialization complex without first having to change your existing programs.



GRS Terminology

- GQSCAN Service provided by GRS to allow tasks to query the status of GRS managed resources and their users.
- Ring Mode Mode of GRS processing that maintains a ring topology of the complex. Each system reviews all the requests for all global resources in the complex.
- Star Mode Mode of GRS processing that maintains a star topology for the complex. Requests for global resources are managed through a lock structure in a coupling facility.

Figure 63. GRS ring and star modes

3.1.3 GRS ring and star modes

Prior to OS/390 Release 2, the global resource serialization ring was the only method used for serializing requests for global resources. The ring consists of one or more systems connected to each other by communication links. The links are used to pass information about requests for global resources from one system to another in the complex. Requests are made by passing a message or token, called the ring system authority message (RSA-message), between systems in a round-robin or ring fashion.

When a system receives the RSA, it inserts global ENQ and DEQ information and passes it along to the next system to copy. It also makes a copy of the global ENQ/DEQ information that was inserted by other systems. When the RSA returns to the originating system, it knows that all other systems have seen its request, so the request is removed. The system can now process those requests by adding them to the global resource queues, and can now determine which jobs own resources, and which jobs must wait for resources owned by other jobs. Each system takes this information and updates its own global queues.

The overhead of the *ring* structure increased because each system in the ring had to keep information relating to all global resource requests, for all systems in the ring, therefore affecting real storage. Response time was affected due to the cyclic updating of each system with global resource requests, and the interdependency of each system within the ring lead to the development of the *star* method. It should be noted, that the ring method does not require a coupling facility.

OS/390 Release 2 introduced the star method of serializing global resources. The star configuration is built around a Coupling Facility, which is where the global resource serialization lock structure, ISGLOCK resides. In a star complex, when an ENQ, DEQ, or RESERVE macro is issued for a global resource, MVS uses information in the ISGLOCK structure to coordinate resource allocation across all systems in the sysplex.

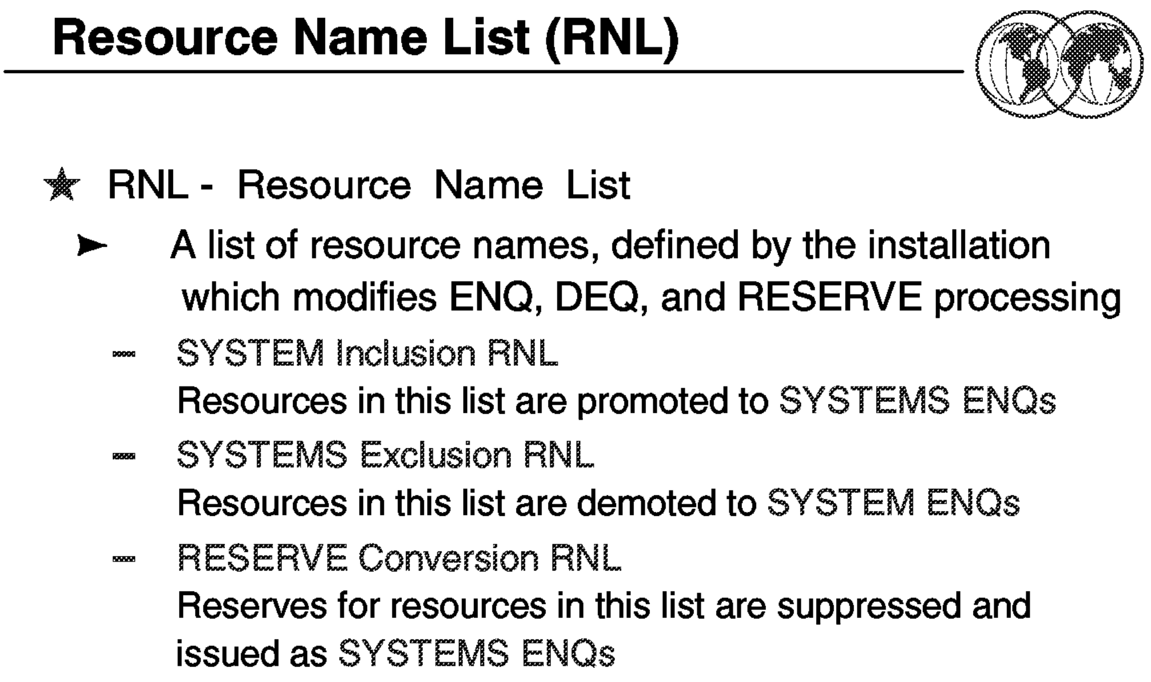
In a star complex, global resource serialization requires the use of the Coupling Facility. The Coupling Facility makes data sharing possible by allowing data to be accessed throughout the sysplex by ensuring the data remains consistent among each system in the sysplex. It should be noted that if you only have one Coupling Facility, using the *star* method is not recommended. Loss of that Coupling Facility would cause all systems in the complex to go into a wait state.

One major purpose of global resource serialization is to eliminate the need to protect data sets on shared DASD volumes by issuing a RESERVE macro that causes a reserve of the entire volume. In general, you want to convert reserves to minimize problems your current methods of resource protection cause. These problems are:

- Interlocks
- Contention between jobs for the same volume
- The possibility that one system might monopolize a shared device
- The data integrity exposure that occurs as a result of a system reset while a reserve is in effect

Global resource serialization counts the number of ENQ/RESERVE requests and the number of pending GQSCAN requests issued by all tasks in each address space. Each time a user issues an ENQ/RESERVE, global resource serialization increases the count in that address space for each resource name and decreases the count when a user in that address space issues a DEQ. Similarly, when a user issues a GQSCAN request, global resource serialization increases the count in that address space and decreases count when the scan is completed (if resumption is requested). As each ENQ, RESERVE, and GQSCAN request is received, global resource serialization determines if increasing the count will exceed the allowed threshold value, and will issue a related abend or return code. This enables GRS to manage the requestor queue for exclusive use of resources.

When using shared DASD, we recommend that your application issue ENQ SCOPE=SYSTEMS and define the resource name on which you are serializing, rather than use the RESERVE macro.



Resource Name List (RNL)

- ★ RNL - Resource Name List
  - ➤ A list of resource names, defined by the installation which modifies ENQ, DEQ, and RESERVE processing
    - SYSTEM Inclusion RNL
      Resources in this list are promoted to SYSTEMS ENQs
    - SYSTEMS Exclusion RNL
      Resources in this list are demoted to SYSTEM ENQs
    - RESERVE Conversion RNL
      Reserves for resources in this list are suppressed and issued as SYSTEMS ENQs

Figure 64. Resource name list (RNL)

3.1.4 Resource name list (RNL)

GRS allows an installation to modify the scope on the ENQ/DEQ and RESERVE processing without changing the applications using the RNLs. The RNL processing is not changed with GRS star support. The RNLs are specified in the parmlib member GRSRNLxx and have a GRS complex scope. GRS allows you to change the scope of an ENQ request without having to rewrite any existing programs, by using three resource name lists (RNL), as follows:

SYSTEM INCLUSION RNL Lists resources for ENQ requests with scope SYSTEM to be converted to scope SYSTEMS.

SYSTEMS EXCLUSION RNL Lists resources for ENQ requests with scope SYSTEMS to be converted to scope SYSTEM.

RESERVE CONVERSION RNL Lists resources for RESERVE requests for which the hardware reserve is to be suppressed and the requests are treated as ENQ requests with scope SYSTEMS.

GRS resource names are made up of two parts:

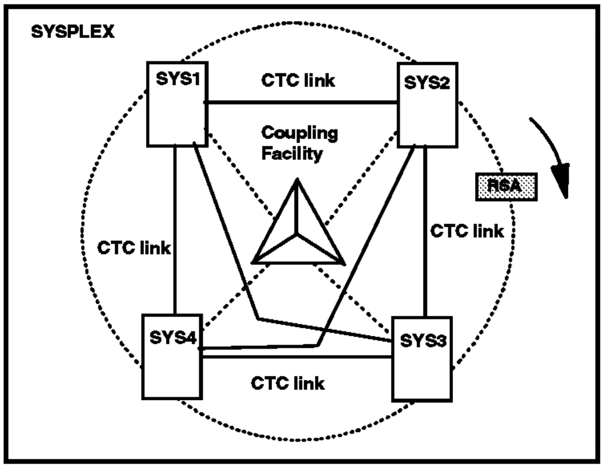
QNAME Specifies an eight-character name. Every request for a serially reusable resource must use the same QNAME, RNAME, and scope to represent a specific resource.

RNAME Specifies the 1 to 255 character name used with QNAME to represent a single resource.

Each RNL entry indicates whether the name is generic or specific. A specific resource name entry matches a search argument only when they are exactly the same. In contrast, a generic resource name entry is a portion of a resource name. A match occurs whenever the specified portion of the generic resource name entry matches the same portion of the input search argument.

Note: RNLs for every system in the GRS complex must be identical; they must contain the same resource name entries, and these must appear in the same order. During initialization, global resource serialization checks to make sure that the RNLs on each system are identical. If they are different, global resource serialization does not allow the system to join the complex.

GRS Ring Topology



- Peer coupling
- Connected via XCF managed path or GRS managed CTC
- Time slicing via RSA
- Each system maintains complex-wide view of data

Figure 65. GRS ring topology

3.2 GRS ring topology

The GRS ring topology has been available since MVS/SP Version 1.3. In the design, the configuration is viewed as a ring of systems.

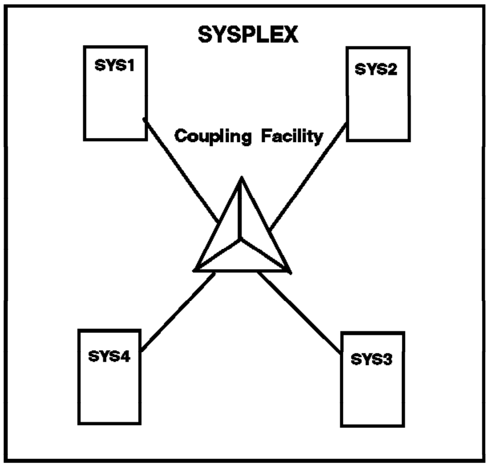
The visual shows an example of a ring complex configuration.

A GRS complex consists of one or more systems connected to each other in a *ring* configuration using:

- XCF communication paths (CTCs) and signalling paths through a Coupling Facility, for the OS/390 systems that belong to a sysplex. If all OS/390 systems belong to a sysplex, then GRS uses XCF services to communicate along the GRS ring; this configuration is called a sysplex matching complex.
- GRS managed channel-to-channel (CTCs) adapters; for the OS/390 systems that do not belong to a sysplex, this configuration is called a mixed complex.

Logically, GRS communication follows a sequential ring philosophy, with GRS message traffic flowing to each system in turn. This message traffic consists of a ring system authority (RSA) message, which contains details of all the GRS requests within the GRS complex. There is a single RSA message token that is passed sequentially around the GRS ring, pausing in each system for local processing and being updated with any serialization changes that have occurred since the last pass of the message.

GRS Star Topology



- Peer Coupling
- Connected via Coupling Facility Lock Structure
- Does not use time slicing as GRS Ring does
- Each system maintains local view of data

Figure 66. GRS star topology

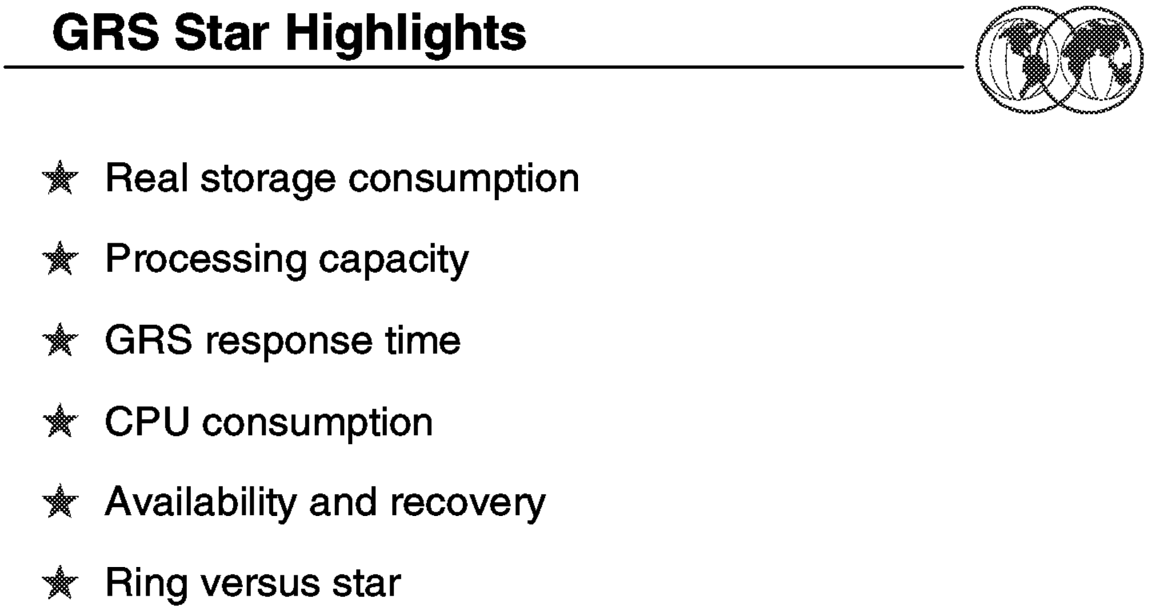
3.3 GRS star topology

With the introduction of the GRS star support, GRS will use the contention detection and management capability of the XES lock structure to determine and assign ownership of a particular global resource. Each system maintains only a local copy of its own global resources, and the GRS Coupling Facility lock structure has the overall image of all system global resources in use.

A GRS star complex configuration uses:

- A Coupling Facility lock structure
- XCF communication paths for global GQSCAN service

The visual shows a GRS star complex, where the global ENQ/DEQ services use the Coupling Facility lock structure to manage and arbitrate the serialization requests, and the global GQSCAN service uses the XCF services to communicate across the sysplex.



GRS Star Highlights

- ★ Real storage consumption
- ★ Processing capacity
- ★ GRS response time
- ★ CPU consumption
- ★ Availability and recovery
- ★ Ring versus star

Figure 67. GRS star highlights

3.4 GRS star highlights

The highlights of the GRS star design are:

- Real storage consumption
- Processing capacity
- CPU consumption
- Availability and recovery
- Ring versus star topology

3.4.1 Real storage consumption

In a GRS ring configuration, each system in the ring maintains a queue of all global resource serialization requests for the entire sysplex. Because of the frequency of ENQ/DEQ processing, each system uses real storage that is proportional to the number of outstanding resource requests and the number of systems in the sysplex. Basically, in a GRS ring the amount of real storage consumed by GRS in each system goes up linearly with the number of systems in the sysplex.

In the case of the GRS star support, no system in the sysplex maintains a complete queue of all the global resource requests for the entire sysplex. Because of this approach, the real storage consumption by GRS for each system is governed only by the number of requests from that system.

3.4.2 Processing capacity

In a GRS ring, all global resource serialization requests are circulated around the ring in a single message buffer called the ring system authority (RSA). Basically, the RSA can be viewed as a floating data area that is shared by all systems in the sysplex, but owned by only one system at a time.

To handle a global resource request, the originating system must place the request in the RSA, and pass it around the ring so all other systems are aware of the request.

Clearly, since the RSA is a shared data area of limited size, the number of requests that each system can place in the RSA diminishes as the number of systems sharing the RSA increases. Also, as the number of systems in the ring increases, the length of time for the RSA to be passed around the ring increases. The net effect of this is that the GRS processing capacity goes down as the number of systems in the ring is increased (that is, fewer requests per second are handled by the ring).

In the case of the GRS star support, this problem is eliminated since all of the requests are handled by mapping ENQ/DEQ requests to XES lock structure requests. With this approach, there is no need to wait for the serialization mechanism, the RSA, to be held by this system before processing the global resource request. Being able to process requests as they are received will improve processing capacity. For instance, there can never be a situation where the system will have to defer excess requests (when the RSA is full).

3.4.3 GRS response time

Very closely related to the capacity problem is the response time that it takes for GRS to handle a request. As can be seen from the capacity discussion, an increase in the number of systems in the GRS ring results in an increase in the average length of time a given system must wait before it can receive control of the RSA to initiate a new request. Additionally, the length of time that it then takes to circulate the request around the ring before it can be processed is also increased.

As a result, the GRS response time on a per request basis increases linearly as the number of systems in the ring is increased. In the case of the GRS star support, the same processing flow that addressed the capacity constraint also addresses the problem of responsiveness.

3.4.4 CPU consumption

In a GRS ring, each system must maintain a view of the entire global resource queue. This means each system must not only process the requests originating on it, but also must process the global requests originating on all the other systems in the ring. The effect of this is that each global ENQ/DEQ request generated in a ring consumes processor time on all the systems in the sysplex, and as the sysplex grows, the amount of processor time GRS ring consumes across the sysplex grows.

In the case of the GRS star support, the overhead of processing an ENQ/DEQ request is limited to only the system on which the request originates. Thus, the total processor time consumed across the sysplex will be less than that consumed by a GRS ring.

3.4.5 Availability and recovery

In a GRS ring, if one of the systems fails, all the other systems are affected during the ring reconstruction time. None of the remaining systems in the sysplex can process any further ENQ/DEQ requests until the ring is reconstructed.

To rebuild the ring, each of the remaining systems must resynchronize and rebuild their view of the global resource queue. The amount of time this may take depends on the number of systems in the sysplex and the number of requests in the global resource queue.

In the case of the GRS star support, availability is improved by the fact that there is less interdependency between the systems in the GRS complex. Whenever a system fails, no action need be taken by the other systems in the sysplex, because there is no resource request data kept on any other system for the requesters on the failing system.

All that needs to occur is the analysis of the contention for the resource, which could determine a new owner (or owners) of the resource, should the resource have been owned by the requester on the failing system.

This approach allows for a much quicker recovery time from the systems remaining in the sysplex, since there is no longer a need to clean up the resource queues or determine a new ring topology for the remaining systems.

3.4.6 Ring versus star topology

When choosing a topology configuration, the following considerations apply:

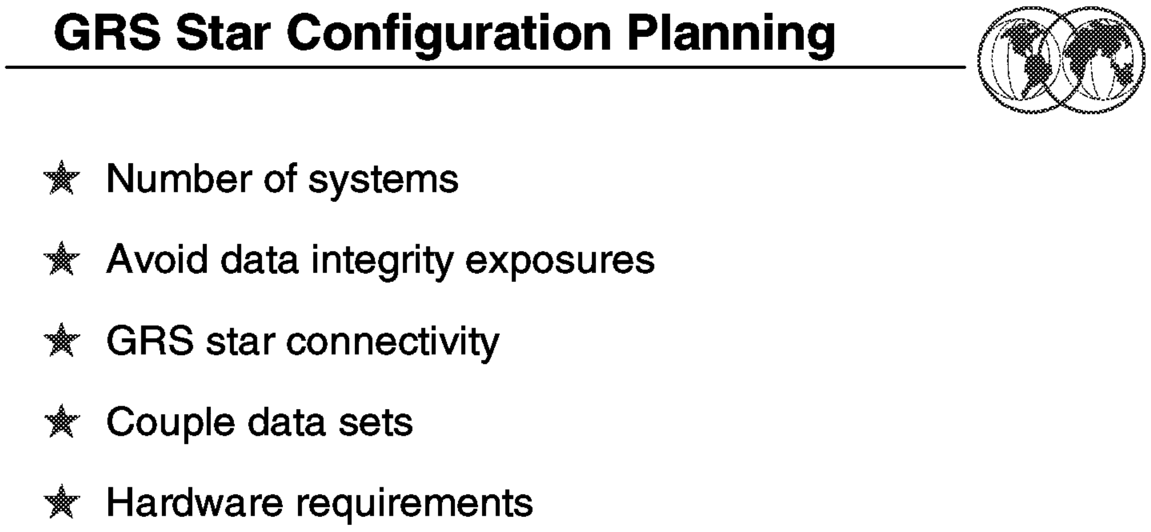
The GRS ring configuration is required when you:

- Have no Coupling Facility available
- Have a mixed ring configuration, with systems that do not belong to the sysplex participating in the GRS complex

The GRS star configuration is suggested for all Parallel Sysplex configurations. The GRS star configuration is recommended when you have a:

- New installation and a Coupling Facility is available
- Large complex of four or more systems
- Heterogeneous set of machines

Note: The GRS star configuration allows for sysplex growth, and also is of value to installations currently running a sysplex because of the improved responsiveness, the reduced consumption of processor and storage, and the better availability and recovery time.



GRS Star Configuration Planning

- Number of systems
- Avoid data integrity exposures
- GRS star connectivity
- Couple data sets
- Hardware requirements

Figure 68. GRS star configuration planning

3.5 GRS star configuration planning

The design of the global resource serialization allows for an unlimited number of systems. The star method for serializing global resources can support complexes of up to 32 systems efficiently and effectively.

Because it is possible for a single installation to have two or more global resource serialization complexes, each operating independently, it should be considered that the independent complexes cannot depend on GRS to share resources. Therefore, there should be no common links made available to GRS star on any two complexes.

3.5.1 Data integrity

To avoid a data integrity exposure, ensure that no system outside the complex can access the same shared DASD as any system in the complex. If that is unavoidable, as often happens, you must serialize the data on the shared DASD with the RESERVE macro. The sample GRS exit ISGGREX0 given in Appendix B, “ISGGREX0 sample exit” on page 349 may help in managing DASD sharing among and outside sysplexes.

As stated before, in a star complex, GRS uses the lock services of the cross-system extended services (XES) component of MVS to serialize access to global resources, and the GRS star method for processing global requests operates in a sysplex like any other MVS component that uses the

Coupling Facility. Therefore, a Coupling Facility with connectivity from all systems in the complex is required.

3.5.2 GRS star connectivity

The overall design of GRS star makes the connectivity configuration relatively simple. MVS requires DASD to be shared by all systems in the sysplex for the sysplex couple data set. An alternate data set is recommended to be used to facilitate the migration from a ring to a star complex, and for availability reasons.

MVS stores information related to sysplex, systems, and XCF groups (such as global resource serialization), on the sysplex couple data set. GRS star stores RNL information into the sysplex couple data set.

3.5.3 Couple data sets

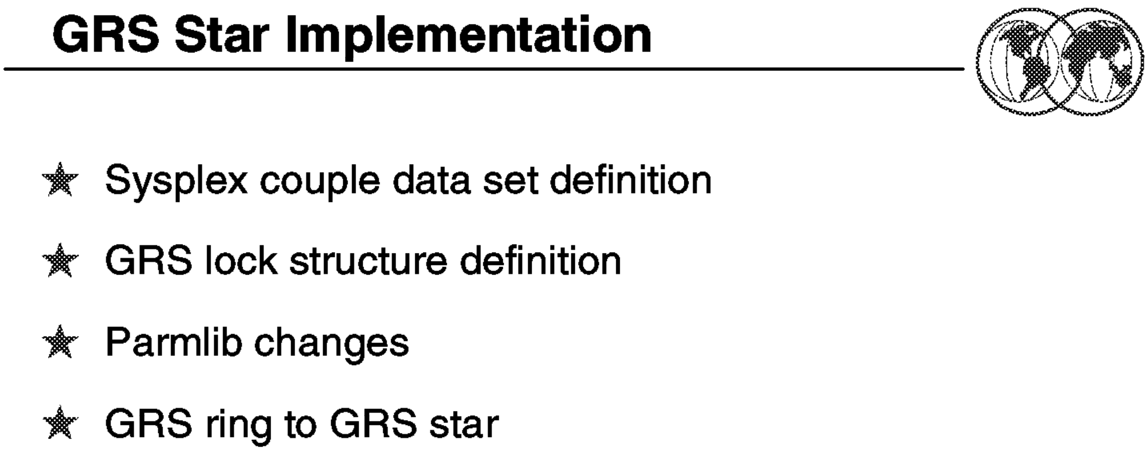
To manage certain aspects of GRS star, the following policies are required:

- Coupling Facility resource management (CFRM) policy, to define the GRS lock structure to MVS
- Sysplex failure management (SFM) policy, to define how MVS is to manage system and signalling connectivity failures

3.5.4 Hardware requirements

In designing a GRS star configuration, verify that the following requirements are met:

- Hardware requirements:
  - A fully interconnected Coupling Facility (accessible by all systems)
  - Coupling Facility links
- Unlike a GRS ring, all systems in a GRS star complex must be in the same sysplex.
- All systems in a star complex must be connected to a Coupling Facility containing the GRS lock structure whose name should be ISGLOCK. For availability reasons in case of Coupling Facility failure, a second Coupling Facility should be available to rebuild the structure used by GRS.
- No channel-to-channel (CTC) connection of systems, other than those managed by XCF, will be supported by GRS. GRS star does not use CTC links.
- A GRS star complex and a ring complex cannot be interconnected, and therefore they cannot coexist in a single GRS complex.
- GRS star does not support systems that are not in the sysplex. A mixed complex is not possible.
- All systems must be at OS/390 Release 2 level or above.



GRS Star Implementation

- Sysplex couple data set definition
- GRS lock structure definition
- Parmlib changes
- GRS ring to GRS star

Figure 69. GRS star implementation

3.6 GRS star implementation

The visual shows the steps necessary to implement the GRS star complex. It is assumed that a sysplex has been already implemented, that a Coupling Facility is operational, and that a GRS ring complex that matches the sysplex is working. This also means that the RNLs have been implemented according to the installation needs.

1. Before switching GRS to a star configuration, all MVS systems must be at OS/390 Release 2 or higher.

2. If applicable, convert from mixed GRS complex (complex not matching the sysplex) to a "pure" sysplex GRS complex.

3.6.1 Define sysplex couple data set

Add a GRS record to the sysplex couple data sets for the star complex. A couple data set must be formatted with the new GRS parameter before initializing a star complex or migrating to a star sysplex from a ring complex. The GRS parameter is in the DEFINEDS statement for the sysplex couple data set. The IXCL1DSU utility must be used; if not, the following diagnostic messages are issued:

```
                  DATA TYPE(SYSPLEX)
                       ITEM NAME(GRS) NUMBER(1)
IXC291I ITEM NAME NOT DEFINED, GRS
```

For more information on the IXCL1DSU utility, see *OS/390 MVS Setting Up a Sysplex*, GC28-1779.

The DEFINEDS statement for the TYPE(SYSPLEX) couple data set is as follows:

```
DEFINEDS SYSPLEX(WTSCPLX1)
     DSN(SYS1.XCF.CDS01) VOLSER(TOTDS1)
     MAXSYSTEM(16)
     CATALOG
   DATA TYPE(SYSPLEX)
   ITEM NAME(GRS) NUMBER(1)
   ITEM NAME(GROUP)  NUMBER(100)
   ITEM NAME(MEMBER) NUMBER(200)
```

The ITEM NAME(GRS) NUMBER(1) record allocates storage in the couple data set for use by GRS and is used to maintain the RNL. The support for star complex *does not* introduce changes in RNL functions or in the dynamic RNL changes.

Note: For all couple data sets, it is recommended you allocate and format a second couple data set to be used as an alternate, for availability reasons.

3.6.1.1 Make CDS available to sysplex

Use the SETXCF command if the sysplex is already active. The following is an example of the SETXCF operator commands:

SETXCF COUPLE,PSWITCH Required if an alternate is active. Switches the current alternate sysplex couple data set to become the primary sysplex couple data set. This command removes the primary old sysplex couple data set from the service.

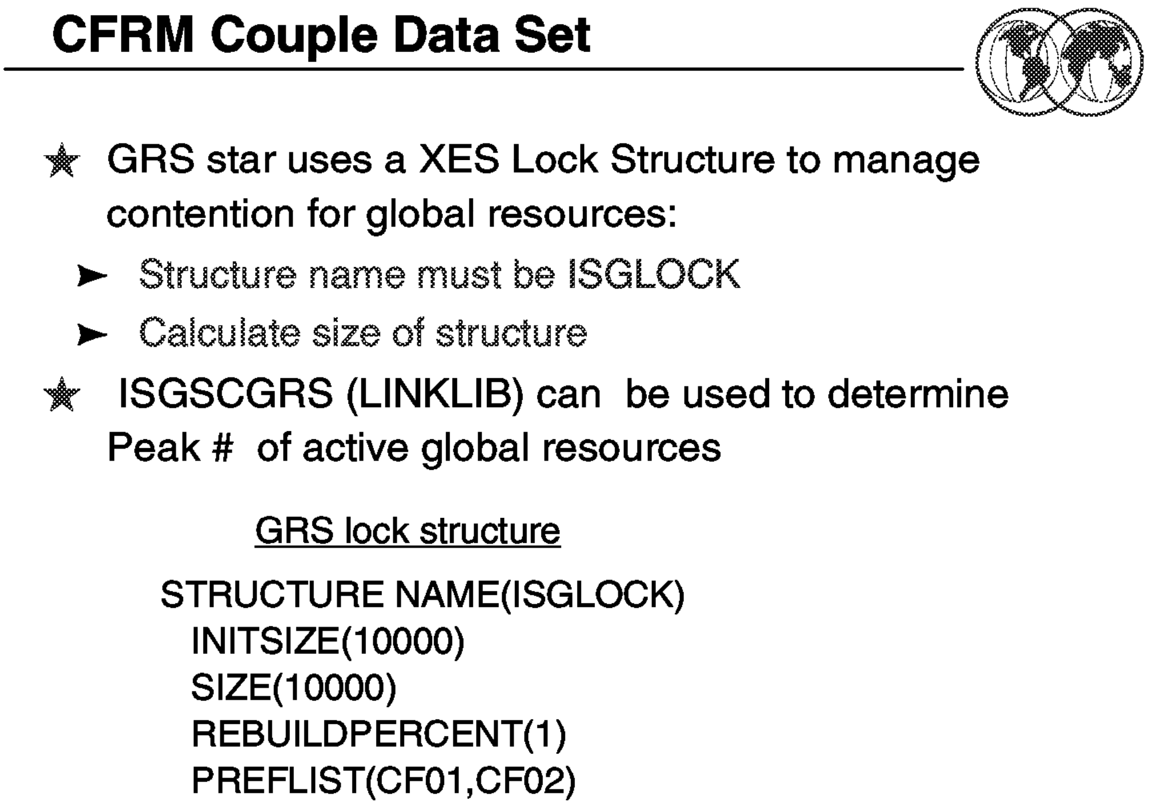
SETXCF COUPLE,ACOUPLE=dsname1 Specifies the data set to be used as an alternate sysplex couple data set.

SETXCF COUPLE,PSWITCH Switches the current alternate sysplex couple data set to become the primary sysplex couple data set. This command removes the primary old sysplex couple data set from the service.

SETXCF COUPLE,ACOUPLE=dsname2 Add dsname2, newly formatted data set, as alternate couple data set.

The COUPLExx parmlib member defines the sysplex couple data set names:

```
COUPLE SYSPLEX(WTSCPLX1)
      PCOUPLE(SYS1.XCF.CDS01)
      ACOUPLE(SYS1.XCF.CDS02)
```



CFRM Couple Data Set

- GRS star uses a XES Lock Structure to manage contention for global resources:
  - Structure name must be ISGLOCK
  - Calculate size of structure
- ISGSCGRS (LINKLIB) can be used to determine Peak # of active global resources

GRS lock structure

```
STRUCTURE NAME(ISGLOCK)
  INITSIZE(10000)
  SIZE(10000)
  REBUILDPERCENT(1)
  PREFLIST(CF01,CF02)
```

Figure 70. CFRM couple data set

3.6.2 Define GRS lock structure

The visual shows an example GRS lock structure definition in the CFRM policy. Use the IXCMIAPU utility to define this structure.

For more information on the IXCMIAPU utility, see *OS/390 MVS Setting Up a Sysplex*, GC28-1779. GRS uses the XES lock structure to reflect a composite system level of interest for each global resource. The interest is recorded in the user data associated with the lock request.

The name of the lock structure must be ISGLOCK, and the size depends on the following factors:

- The size and type of systems in the sysplex
- The type of workload being performed

GRS requires a lock structure large enough for at least 32K entries. The number of lock entries used depends on the number of outstanding global requests. In a typical environment, the major contributors for active global requests are related to the number of data sets and databases allocated (qname SYSDSN), VSAM data sets opened (qname SYSVSAM), and ISPF-used data sets (qname SPFEDIT). It is assumed that the major name SYSDSN is in the RNL inclusion list.

A small sysplex, made up of smaller processors and running a transaction processing workload, needs a smaller structure than a larger sysplex that is composed of large processors and running a batch and TSO workloads combination.

It is recommended you use the following formula to determine the size for ISGLOCK:

```
# lock table entries = peak # resources * 100

Round # lock table entries up to next power of 2

Lock table size (in bytes) = # lock table entries * 8

Structure Size (in K bytes) = (lock table size/1024) + 256
```

Where:

Peak # Is the number of unique globally-managed resources (SYSTEMS ENQs and converted RESERVEs) measured at a time of peak system utilization.

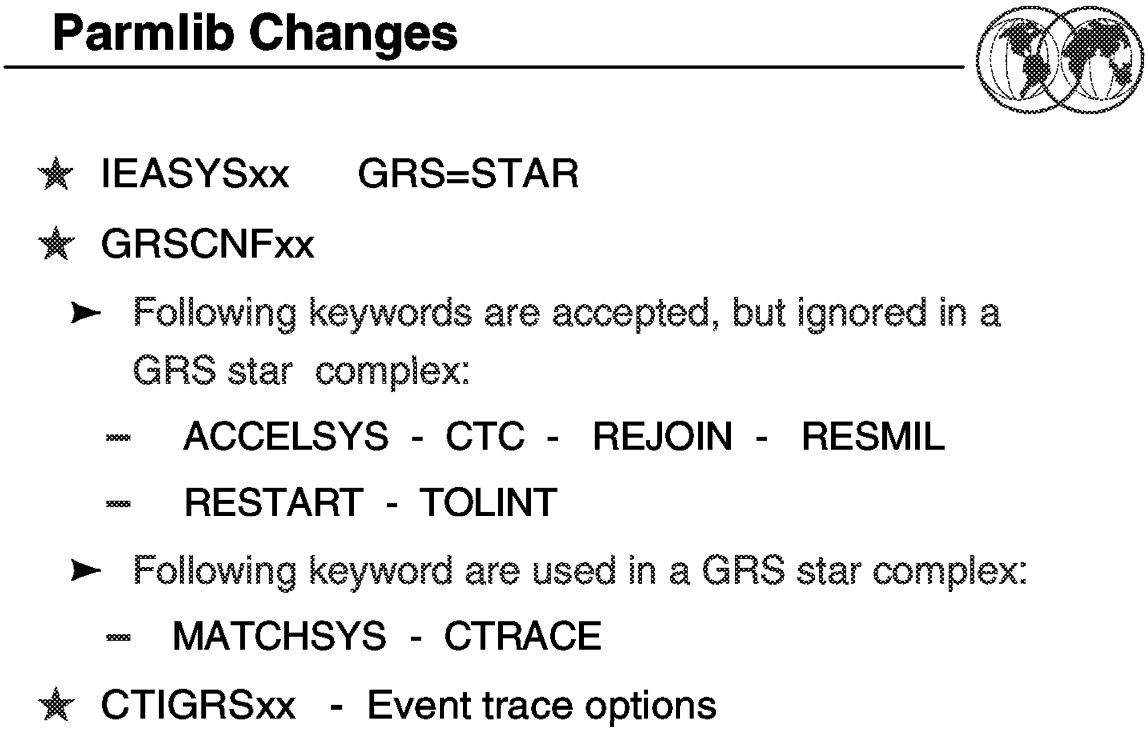
IBM has provided a program named ISGSCGRS in SYS1.LINKLIB, that runs GQSCAN. ISGSCGRS returns the number of global resources in the complex.

ISGSCGRS can be used to verify the number of outstanding global requests. The program issues a WTO every 10 seconds indicating the number of outstanding requests and runs for two minutes. To execute the program, use the following JCL, which can be found in SYS1.SAMPLIB member ISGCGRS:

```
//ANYNAME  JOB (999,POK),CLASS=A,MSGCLASS=X,TIME=1440
//STEP001 EXEC PGM=ISGSCGRS
```

An example of the issued WTOs is:

```
 ANYNAME  00000090  +Number of global resources outstanding: 000022253
```



Parmlib Changes

- IEASYSxx GRS=STAR
- GRSCNFxx
  - Following keywords are accepted, but ignored in a GRS star complex:
    - ACCELSYS - CTC - REJOIN - RESMIL
    - RESTART - TOLINT
  - Following keyword are used in a GRS star complex:
    - MATCHSYS - CTRACE
- CTIGRSxx - Event trace options

Figure 71. Parmlib changes

3.6.3 Parmlib changes

Initializing a star complex requires specifying STAR on the GRS= system parameter in the IEASYSxx parmlib member, or in response to message IEA101A during IPL. The START, JOIN, and TRYJOIN options apply to a ring complex only.

The PLEXCFG parameter should be MULTISYSTEM; the relation between GRS= and PLEXCFG parameters are:

- Use GRS=NONE and PLEXCFG=XCFLOCAL or MONOPLEX when there is a single-system sysplex and no GRS complex.
- Use GRS=TRYJOIN (or GRS=START or JOIN) and PLEXCFG=MULTISYSTEM when there is a sysplex of two or more systems and the GRS ring complex uses XCF signalling services.
- Use GRS=STAR and PLEXCFG=MULTISYSTEM when there is a GRS star complex.

3.6.3.1 Parmlib member GRSCNFxx with star complex

Basically, the GRSCNFxx of parmlib is not required when initializing a star complex that uses the default CTRACE parmlib member, CTIGRS00, which is supplied with the system.

If you want to initialize a star complex using a CTRACE parmlib member other than the default, you must use the GRSDEF statement in the GRSCNFxx parmlib member. The following is a GRSCNFxx example that refers to the GRS configuration shown in Figure 66 on page 104:

```
GRSDEF MATCHSYS(*)       /* GRSDEF FOR SYSTEMS SYS1, SYS2, SYS4
       CTRACE(CTIGRS01) /* PARMLIB MEMBER CTIGRS01 CONTAINS TRACE
                        /* OPTIONS
GRSDEF MATCHSYS(SYS3)    /* GRSDEF FOR SYSTEM SYS3
       CTRACE(CTIGRS03) /* PARMLIB MEMBER CTIGRS03 CONTAINS TRACE
                        /* OPTION
```

3.6.3.2 GRS ring to star conversions

Apart from the two GRSDEF parameters listed in the example, all the other parameters on the GRSDEF statement (ACCELSYS, RESMIL, TOLINT, CTC, REJOIN, and RESTART) apply only to systems initializing in a ring complex. Although they can be specified on the GRSDEF statement and are parsed and syntax checked, they are not used when initializing systems into a GRS star complex. Global resource serialization ignores the parameters that are not applicable on the GRSDEF statement when initializing systems into a star complex, as well as when initializing systems into a ring complex.

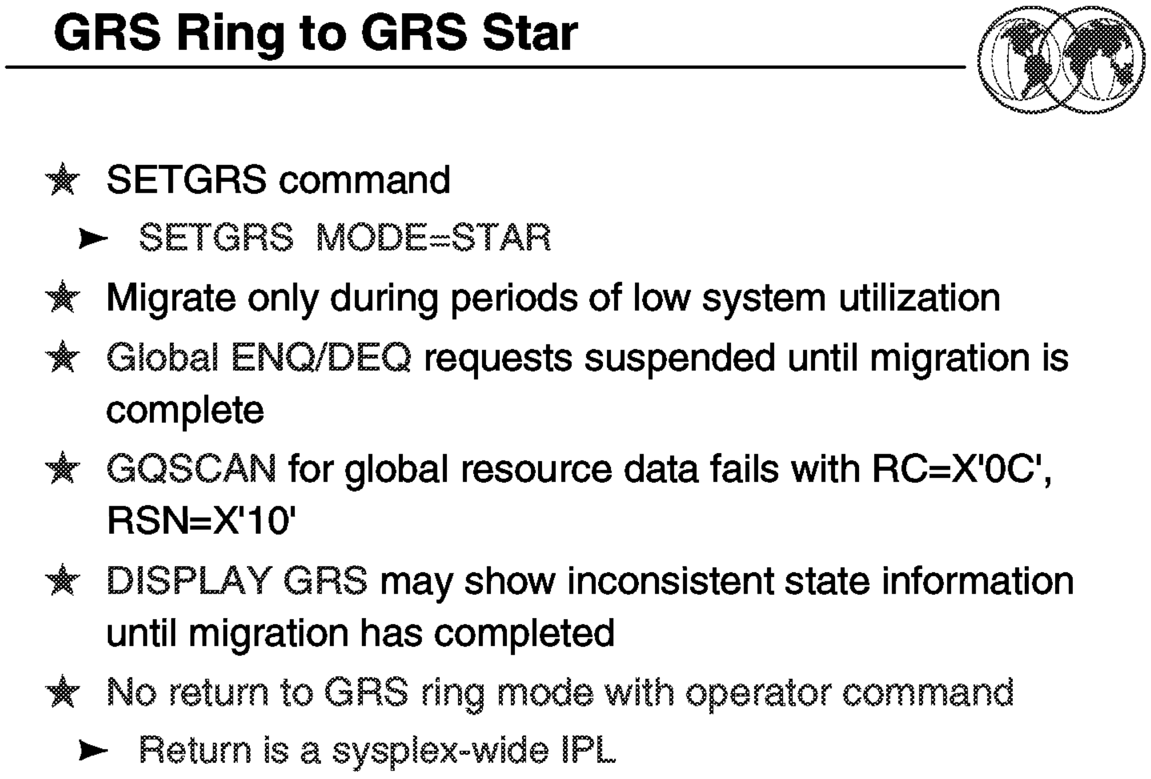
If the ring parameters are left in the GRSCNFxx parmlib member, there is the potential, due to making an error in the specification of the GRS= parameter in IEASYSxx, to create two GRS complexes.

For this reason, even if it is possible for an installation to create a single GRSCNFxx parmlib member that can be used for the initialization of either a star or a ring complex, it is suggested you have different GRSCNFxx members—one for star and one for ring, and use the GRSCNF= keyword in IEASYSxx to select the proper one. This helps in the transition from a ring to a star complex if the installation elects to use the SETGRS MODE=STAR capability to make the transition.

3.6.3.3 Event trace options

Use the CTRACE parameter in GRSCNFxx to specify the CTRACE options you want for your installation. The CTRACE parameter in GRSCNFxx allows you to modify the default tracing options used by global resource serialization. Since the defaults provide adequate serviceability information, you should change them only upon the recommendation of your IBM service representative.

Trace options are provided to select which events should be traced. The options which pertain to the star processing environment are CONTROL, REQUEST, MONITOR, SIGNAL, and FLOW. Each can be made more specific by specifying CONTROLn, REQUESTn, MONITORn, and SIGNALn, or FLOWn to provide greater flexibility and granularity in selecting which events should be traced. When viewing the trace data using the IPCS CTRACE subcommand, filtering options are provided to limit the scope of the data displayed, and subheaders are shown which briefly describe each trace key.



GRS Ring to GRS Star

- ★ SETGRS command
  - ➤ SETGRS MODE=STAR
- ★ Migrate only during periods of low system utilization
- ★ Global ENQ/DEQ requests suspended until migration is complete
- ★ GQSCAN for global resource data fails with RC=X'0C', RSN=X'10'
- ★ DISPLAY GRS may show inconsistent state information until migration has completed
- ★ No return to GRS ring mode with operator command
  - ➤ Return is a sysplex-wide IPL

Figure 72. GRS ring to GRS star

3.6.4 GRS ring to GRS star

The SETGRS command is used to migrate an active ring to a star complex. There is no SETGRS option to migrate from a star to a ring complex; returning to a ring complex requires an IPL of the entire sysplex.

The following command is used to migrate from a ring complex to a star complex:

```
SETGRS MODE=STAR
```

The SETGRS command can be issued from any system in the complex and has sysplex scope.

While processing a SETGRS MODE=STAR command, processing is suspended for the global resource serialization ENQ, DEQ, RESERVE, and GQSCAN for global resource data fails with RC=X′0C′, RSN=X′10′. The length of time global resource serialization requesters are suspended may be a few minutes while the ISGLOCK lock structure and global resource serialization sysplex couple data set records are initialized, and changes to the internal GRS control block structures are initialized as well.

Note: The migration should be invoked at a time when the amount of global resource request activity is likely to be minimal.

The SETGRS MODE=STAR request is valid under the following conditions:

- GRS is currently running a ring complex that exactly matches the sysplex.
- All systems in the ring complex support OS/390 Release 2 or later.
- All systems in the ring complex are connected to a Coupling Facility.

- All systems can access the ISGLOCK lock structure on the Coupling Facility.
- The global resource serialization records are defined on the sysplex couple data sets.
- There are no active dynamic RNL changes in progress.

The DISPLAY GRS (D GRS) command shows the state of each system in the complex. The D GRS shows system status only as it relates to the global resource serialization ring. D GRS does not reflect how well a system is running.

You can also use D GRS to display the local and global resources requested by the systems in the ring, contention information, the contents of the RNLs, and jobs that are delaying or suspended by a SET GRSRNL command.

You can issue D GRS from any system in the ring and at any time after the ring has been started. The D GRS display shows the status of the ring from that system′s point of view; thus, the displays issued from different systems might show different results.

GRS Star Complex Overview

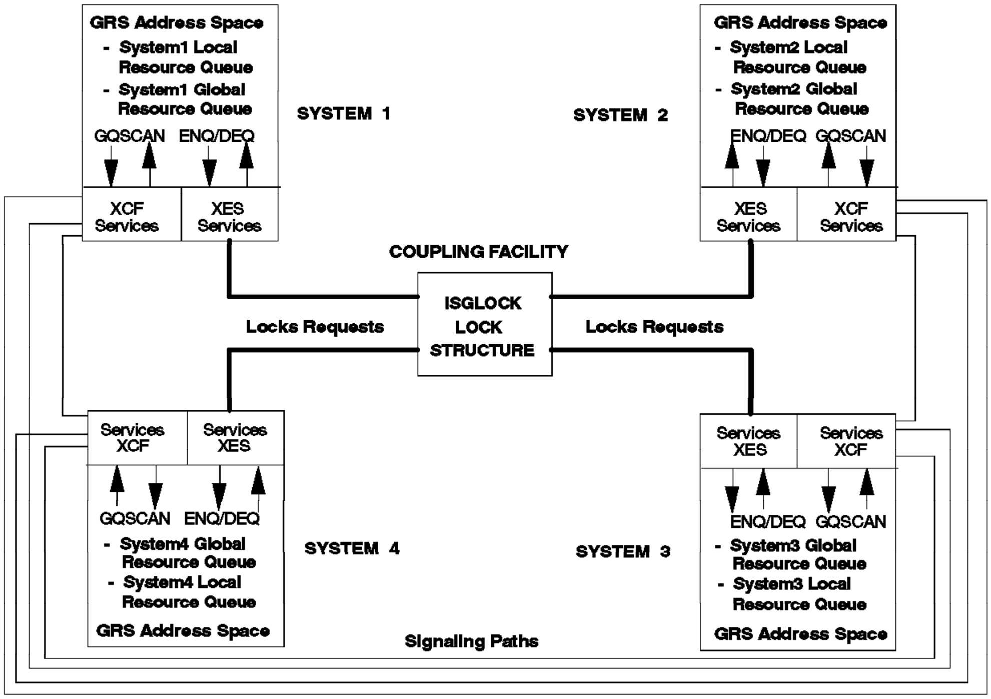


Figure 73. GRS star complex overview

3.7 GRS star complex overview

In a GRS star complex, when an ENQ, RESERVE, or DEQ request is issued for a global resource, the request is converted to an XES lock request. The XES lock structure coordinates the requests it receives to ensure proper resource serialization across all systems in the complex, and notifies the originating systems about the status of each request. Based on the results of these lock requests, GRS responds to the requester with the result of the global request.

As shown in the visual, each system in the GRS star complex has a *server task* (dispatchable unit) executing in the GRS address space. All ENQ, DEQ, RESERVE, and GQSCAN requests for global resources that are issued from any of the address spaces on a system are always first passed to the GRS address space on the requester′s system. In the GRS address space, the server task performs the processing necessary to handle a given request and interfaces with the XES lock structure, if necessary, to process the request. In case of a GQSCAN request for global resource information, the server task packages the request for transmission and sends it to each of the other server tasks in the GRS complex using the XCF services. This is necessary because the status of global requests is maintained at the system level.

Global ENQ Processing

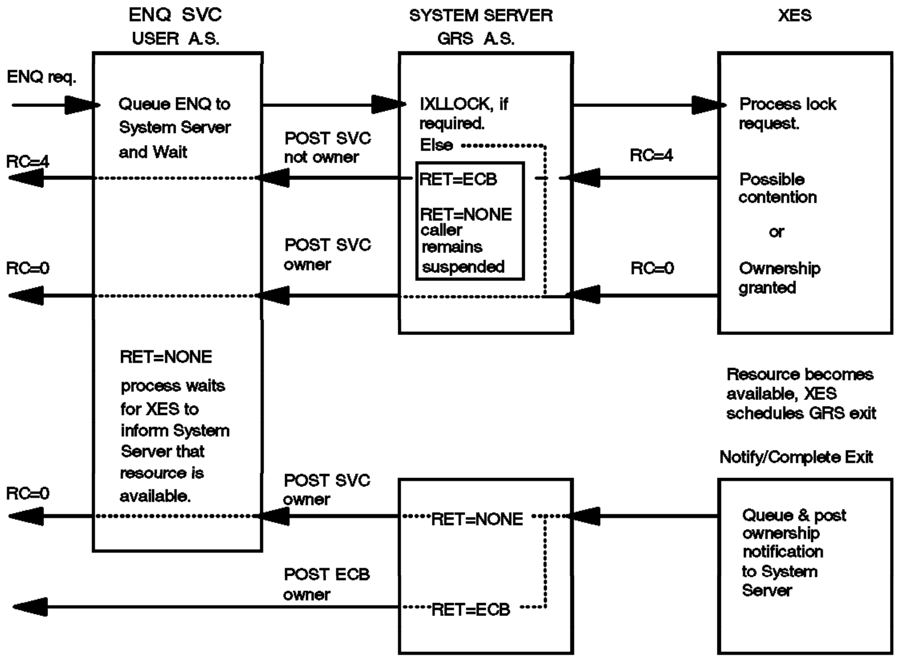


Figure 74. Global ENQ processing

3.7.1 Global ENQ processing

ENQ processing for the RET=NONE and RET=ECB keywords is shown at a high level in the visual. In a GRS star complex, no system maintains a complete view of the outstanding global resource requests, in contrast with the GRS ring philosophy of maintaining a complete view of the global resource requests on all systems in the complex. Instead, each system maintains a queue of each of the local requests, called the *system global resource queue*. Arbitrating requests for global resources from different systems in the complex is managed by putting a subset of the information from the system global resource queue into the user data associated with the lock request made to the XES lock structure for a particular resource.

In a GRS star complex, requests for ownership of global resources are handled through a lock structure on a Coupling Facility that is fully interconnected with all the systems in the sysplex. GRS interfaces with the XES lock structure to reflect a composite system-level view of interest in each of the global resources for which there is at least one requester. This interest is recorded in the user data associated with the lock request.

Each time there is a change in the composite state of a resource, GRS updates the user data reflecting the new state of interest in the XES lock structure. In general, this composite state is altered each time a change is made to the set of owners of the resource or the set of waiters for the resource.

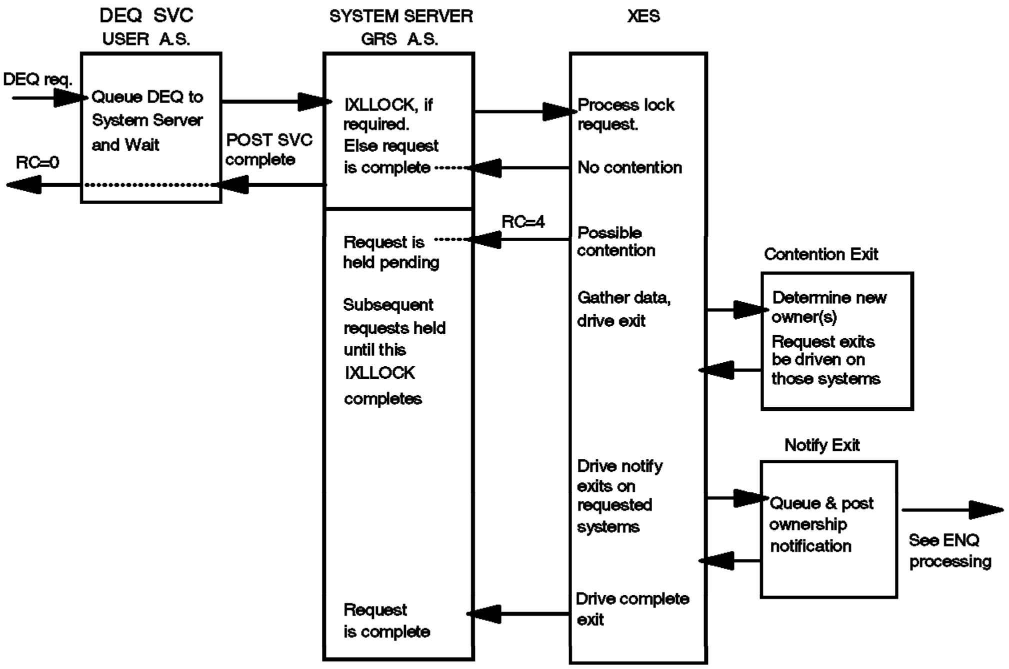
Each time an ENQ request is received, the server task analyzes the state of the resource request queue for the resource. If the new request alters the composite state of the queue, an IXLLOCK request is made to reflect the changed state of the resource for the requesting system. If the resource is immediately available, the IXLLOCK request indicates that the system can grant ownership of the resource.

If the XES lock structure cannot immediately grant ownership of the resource, the IXLLOCK request completes with the return code X′04′, and the server task holds the request pending and the requester in a wait state until the completion exit is driven for the request.

Only one IXLLOCK can ever be in progress from a particular system and for a particular global resource at one time. Requests for a resource that are received while a request is in progress are held pending until all preceding requests for the resource have been through IXLLOCK processing. The requests are processed in first-in first-out (FIFO) order.

If contention exists for the resource, the requester is not granted ownership until some set of requesters dequeue from the resource. When the requester is to be assigned ownership of the resource, the contention exit, driven as the result of a DEQ request, drives the notify exit of the server task that is managing the requester. The notify exit informs the server task of the ownership change. Finally, the server task informs the requester to continue by posting for RET=NONE RC=0 or RC=4 in the the requester′s ECB for RET=ECB. The requester now owns the resource.

Global DEQ Processing



DEQ SVC
USER A.S.

SYSTEM SERVER
GRS A.S.

XES

DEQ req.

Queue DEQ to System Server and Wait

RC=0

POST SVC complete

IXLLOCK, if required.
Else request is complete

Process lock request.

No contention

RC=4

Request is held pending

Subsequent requests held until this IXLLOCK completes

Possible contention

Gather data, drive exit

Contention Exit

Determine new owner(s)
Request exits be driven on those systems

Notify Exit

Drive notify exits on requested systems

Queue & post ownership notification

See ENQ processing

Request is complete

Drive complete exit

Figure 75. Global DEQ processing

3.7.2 Global DEQ processing

In a GRS star complex, releasing a resource has the effect of altering the system′s interest in the resource. GRS alters the user data associated with the resource, and issues an IXLLOCK request to reflect the new interest. The visual illustrates the global DEQ process.

After resuming the user, the server task performs local processing and issues an IXLLOCK request if the DEQ results in an alteration of the system′s composite state of the resource. If there is no global contention on the resource, XES returns from the IXLLOCK service with a return code of X′0′, indicating that no asynchronous ownership processing will occur.

When IXLLOCK completes with return code X′04′, XES indicates that there may be contention for the resource and that asynchronous processing occurs for this resource. The server task places this resource into a pending state, keeping all subsequent requests queued in FIFO order until the asynchronous processing completes.

XES gathers the user data from each of the systems interested in the resource and presents them to the contention exit that GRS provides. The contention exit determines, based on each of the system′s composite states, which systems have owners of the resource. The contention exit causes the notify exit on each of the systems to be driven. The notify exit informs the server task of the change in ownership of the resource. The server task posts the new owner (or owners) of the resource.

In a GRS star complex, a DEQ request is handled as a fast DEQ under the following conditions:

- The DEQ is for a single resource.

- The resource request (ENQ for the resource) has already been reflected in the GRS lock structure.
- The request for the resource is not the target of a MASID/MTCB request.

This is equivalent to the implementation of fast DEQ in a GRS ring complex.

If the resource request has not yet been reflected in the GRS lock structure, the DEQ requester waits until the IXLLOCK for that request has been completed. Following this, the request is resumed in parallel with the completion of the DEQ request.

GQSCAN Request for Global Resource Data

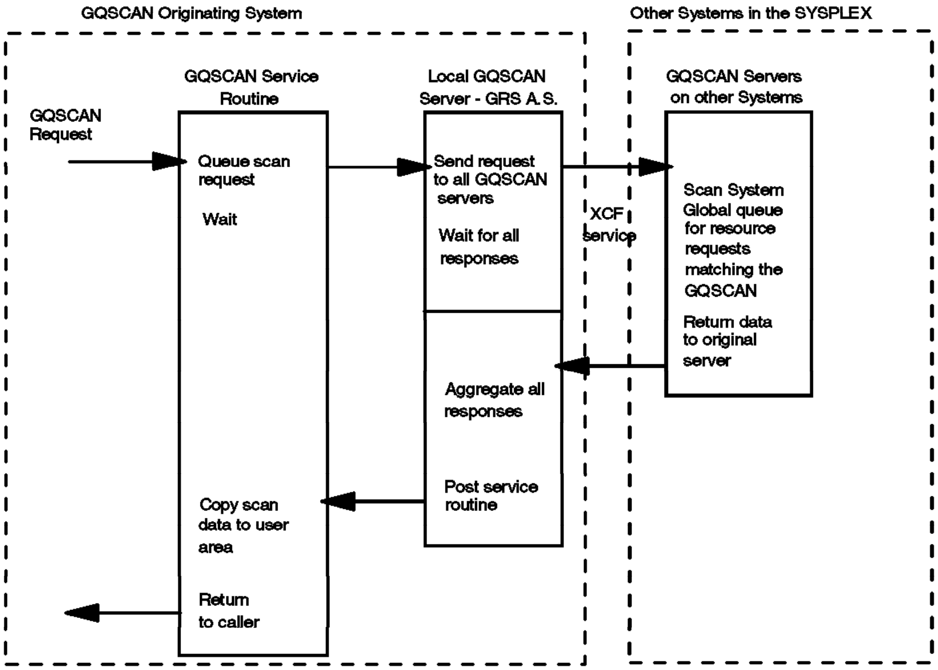

GQSCAN Originating System
Other Systems in the SYSPLEX
GQSCAN Request
GQSCAN Service Routine
Queue scan request
Wait
Copy scan data to user area
Return to caller
Local GQSCAN Server - GRS A.S.
Send request to all GQSCAN servers
Wait for all responses
Aggregate all responses
Post service routine
XCF service
GQSCAN Servers on other Systems
Scan System Global queue for resource requests matching the GQSCAN
Return data to original server

Figure 76. GQSCAN request for global resource data

3.7.3 GQSCAN request for global resource data

The basic flow of a GQSCAN request for global resource in a star complex is shown in the visual.

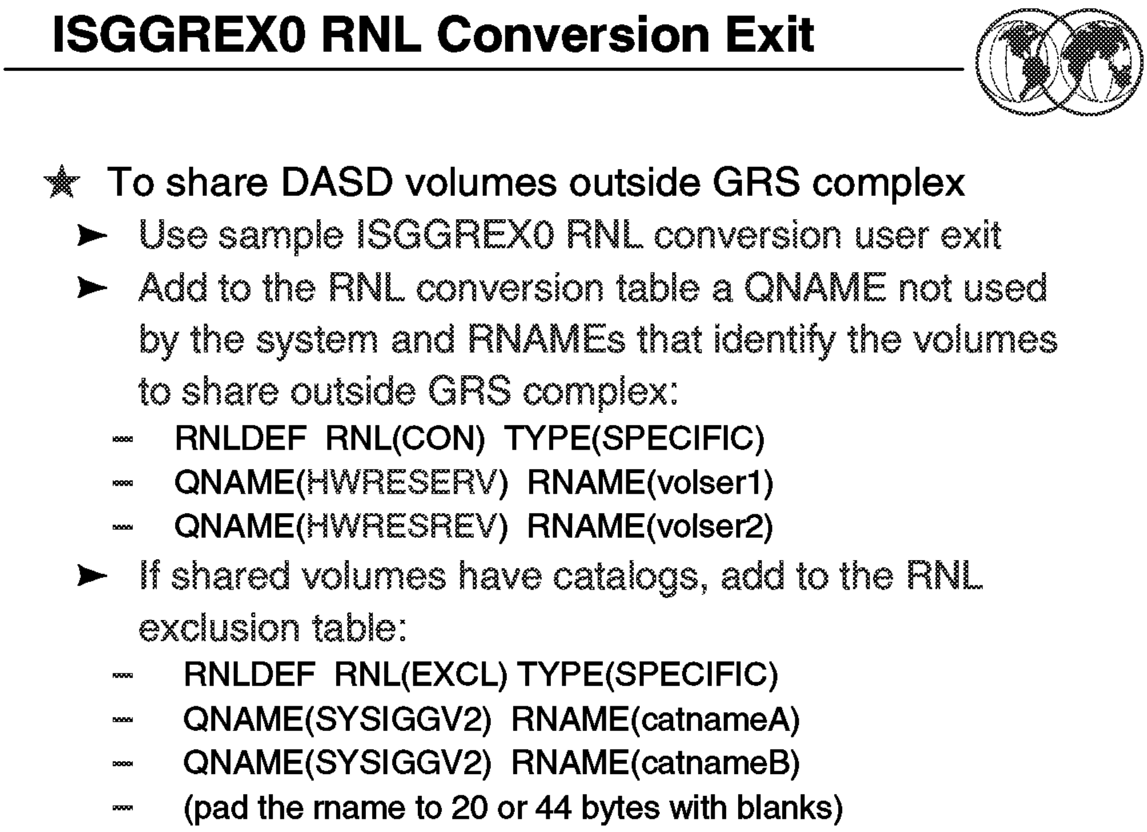
In the case of a GQSCAN request for global resource information, the request is passed from the issuing address space to the GRS server task on that system, and the GQSCAN request is “waited.” The server task then packages the request for transmission and sends it to each of the server tasks in the GRS star complex.

Each server task in the complex scans their system global resource queue that matches the GQSCAN selection criteria, responding to the originating system with the requested global resource information (if any) and a return code. The originating server task waits for responses from all of the server tasks in the complex and builds a composite response to be returned to the caller.

Waiting the issuer of GQSCAN when global resource information is requested is a significant change from the way GQSCAN works in a GRS ring complex. In a GRS ring complex, GQSCAN processing is performed synchronously with respect to the caller′s task.

A new GQSCAN macro option is provided to allow the issuer to indicate whether or not cross-system processing for global resource information should be performed.

The new “no cross-system” option (XSYS=NO) is provided primarily for the GQSCAN that cannot be put into a wait, and does not require that data about requesters on other systems in the complex. XSYS=YES is the default option.



ISGGREX0 RNL Conversion Exit

- ★ To share DASD volumes outside GRS complex
  - ➤ Use sample ISGGREX0 RNL conversion user exit
  - ➤ Add to the RNL conversion table a QNAME not used by the system and RNAMEs that identify the volumes to share outside GRS complex:
    - RNLDEF RNL(CON) TYPE(SPECIFIC)
    - QNAME(HWRESERV) RNAME(volser1)
    - QNAME(HWRESREV) RNAME(volser2)
  - ➤ If shared volumes have catalogs, add to the RNL exclusion table:
    - RNLDEF RNL(EXCL) TYPE(SPECIFIC)
    - QNAME(SYSIGGV2) RNAME(catnameA)
    - QNAME(SYSIGGV2) RNAME(catnameB)
    - (pad the rname to 20 or 44 bytes with blanks)

Figure 77. ISGGREX0 RNL conversion exit

3.8 ISGGREX0 RNL conversion exit

When GRS receives an ENQ/RESERVE/DEQ request for a resource with a scope of SYSTEM or SYSTEMS, unless the request specifies RNL=NO or you have specified GRSRNL=EXCLUDE on IEASYSxx parmlib member, the installation-replaceable scan RNL exit (ISGGREX0) is invoked to search the appropriate RNL. A return code from ISGGREX0 indicates whether or not the input resource name exists in the RNL.

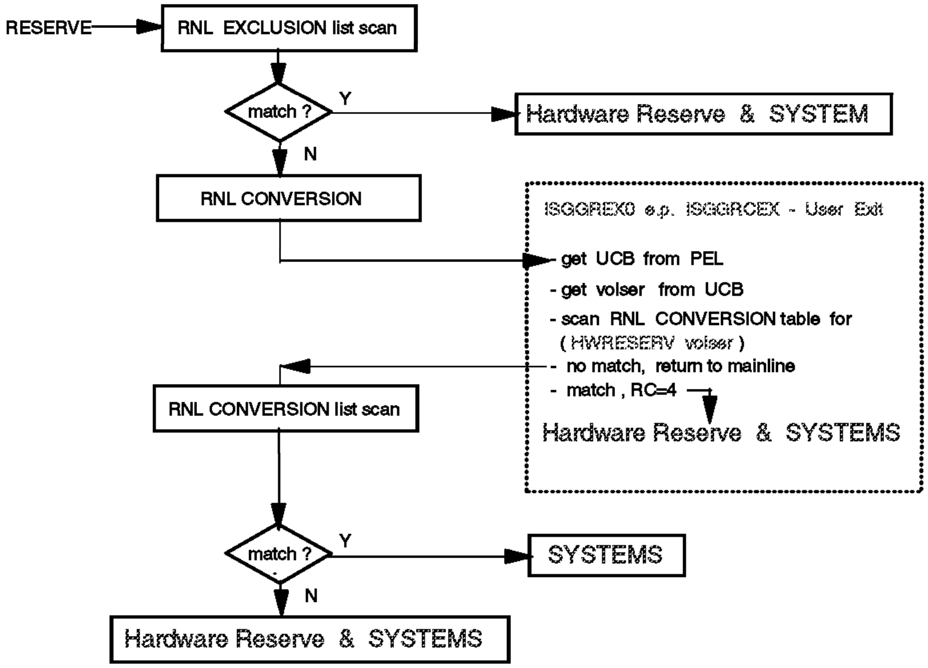
Input to the ISGGREX0 consists of the QNAME, the RNAME, and the RNAME length of the resource. The IBM default ISGGREX0 exit search routine finds a matching RNL entry when:

- A specific resource name entry in the RNL matches the specific resource name in the search argument.

  Note: The length of the specific RNAME is important. A specific entry does not match a resource name that is padded with blank characters to the right.

- A generic QNAME entry in the RNL matches the QNAME of the search argument.
- A generic QNAME entry in the RNL matches the corresponding portion of the resource name in the search argument.

ISGGREX0 Conversion Exit Flow



RESERVE

RNL EXCLUSION list scan

match ?

Y

Hardware Reserve & SYSTEM

N

RNL CONVERSION

ISGGREX0 e.p. ISGGRCEX - User Exit

get UCB from PEL

- get volser from UCB
- scan RNL CONVERSION table for (HWRESERV volser)
- no match, return to mainline
- match , RC=4

Hardware Reserve & SYSTEMS

RNL CONVERSION list scan

match ?

Y

SYSTEMS

N

Hardware Reserve & SYSTEMS

Figure 78. ISSGREX0 conversion exit flow

3.8.1 ISSGREX0 conversion exit flow

GRS has a user exit, ISGGREX0, that receives control whenever an ENQ/DEQ/RESERVE request is issued for a resource. ISGGREX0 scans the input resource name list (RNL) for the resource name specified in the input parameter element list (PEL). A return code from the exit routine indicates whether or not the input resource name appears in the RNL.

You can use ISGGREX0 to determine whether the input resource name exists in the RNL. Replacing ISGGREX0 changes the technique that GRS normally uses to scan an RNL. Changing the search technique can have an adverse effect on system performance, especially when the RNLs contain many specific entries.

The routine has three external entry points:

- ISGGSIEX scans the SYSTEM inclusion RNL
- ISGGSEEX scans the SYSTEM exclusion RNL
- ISGGRCEX scans the RESERVE conversion RNL

Depending on the RNL the request requires, the exit routine is invoked at the appropriate entry point for the SYSTEM inclusion RNL, the SYSTEM exclusion RNL, or the RESERVE conversion RNL.

You can modify the IBM-supplied exit routine to perform the processing necessary for your system. Use the source version of the exit routine provided as member ISGGREXS in SYS1.SAMPLIB.

For example, ISGGRCEX can be used to avoid converting reserves against a volume on a system outside the global resource serialization complex. An example is given in Appendix B, "ISGGREX0 sample exit" on page 349.

Note: The ISGGREX0 exit routine in each system belonging to the same GRS complex must yield the same scan results; otherwise, resource integrity cannot be guaranteed. For the same reason, the RNLs themselves must be the same. The exit routine resides in the nucleus and to change or activate it, a sysplex-wide IPL is required. See *OS/390 MVS: Installation Exits*, SC28-1753, for details.

Chapter 4. OS/390 systems operations

To monitor MVS and to respond to system changes and problems makes operations planning more important than ever before. In order to make decisions about MVS operations planning, you need to understand:

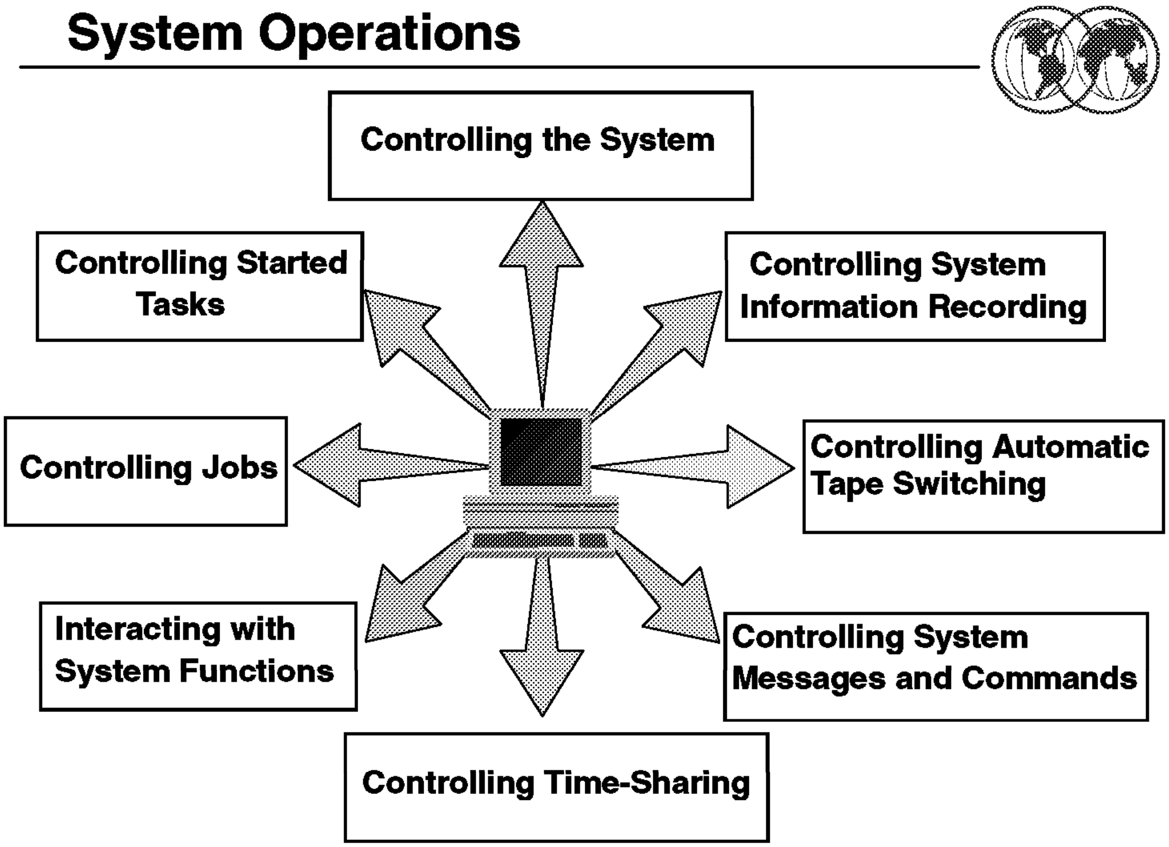
- The operations goals of your installation
- The operating environment and how it will affect those goals

The operation of an MVS system involves the following:

- Console operations or how operators interact with MVS to monitor or control the hardware and software.
- Message and command processing that forms the basis of operator interaction with MVS and the basis of MVS automation.

Operating MVS involves managing hardware like processors and peripheral devices (including the consoles where your operators do their work) and software such as the MVS operating control system, the job entry subsystem, subsystems like NetView that can control automated operations, and all the applications that run on MVS.

Planning MVS operations for a system must take into account how operators use consoles to do their work and how you want to manage messages and commands. Because messages are also the basis of automated operations, understanding message processing in an MVS system can help you plan MVS automation.



System Operations

Controlling the System

Controlling Started Tasks

Controlling System Information Recording

Controlling Jobs

Controlling Automatic Tape Switching

Interacting with System Functions

Controlling System Messages and Commands

Controlling Time-Sharing

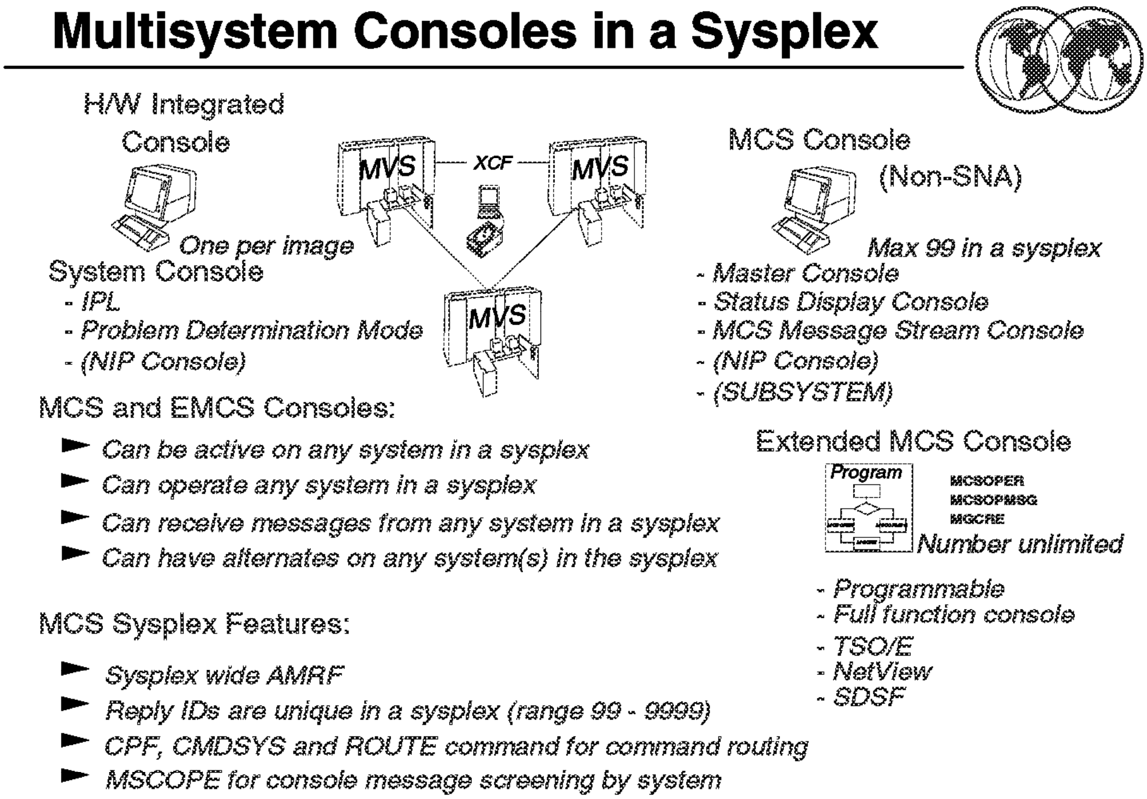
Figure 79. System operations

4.1 OS/390 system operations

Controlling the operating system effectively, includes the following tasks:

- Display current system status, such as the number of active jobs and teleprocessing functions, so you can take appropriate actions to operate the system efficiently and to correct potential problems.
- Display the status of devices and availability of paths.
- Communicate among several consoles.
- Communicate within a sysplex. In a sysplex, several MVS systems function together to process work, and you might need to know about the operations of more than one system in a sysplex.
- Set the time and change the system parameters.
- Use the system restart function to control certain system functions.
- Respond to message IEA502A.
- Respond to message BLW004A.
- Activate a workload management service policy for a sysplex.
- Switch the workload management mode in effect on a system.

MVS provides system and subsystem commands that display job and system status either when requested or continually at a regular interval. Other commands route status information to one or more consoles and provide communication among operators in a multiple-console environment, as well as communication with time-sharing users. Many commands let you display information about all the systems in a sysplex, and some commands allow you to control any target system in the sysplex.



Multisystem Consoles in a Sysplex

H/W Integrated Console

One per image

System Console

- *IPL*
- *Problem Determination Mode*
- *(NIP Console)*

MCS and EMCS Consoles:

- *Can be active on any system in a sysplex*
- *Can operate any system in a sysplex*
- *Can receive messages from any system in a sysplex*
- *Can have alternates on any system(s) in the sysplex*

MCS Sysplex Features:

- *Sysplex wide AMRF*
- *Reply IDs are unique in a sysplex (range 99 - 9999)*
- *CPF, CMDSYS and ROUTE command for command routing*
- *MSCOPE for console message screening by system*

MCS Console (Non-SNA)

Max 99 in a sysplex

- *Master Console*
- *Status Display Console*
- *MCS Message Stream Console*
- *(NIP Console)*
- *(SUBSYSTEM)*

Extended MCS Console

Number unlimited

- *Programmable*
- *Full function console*
- *TSO/E*
- *NetView*
- *SDSF*

Figure 80. Multisystem consoles in a syplex

4.2 Multisystem consoles in a sysplex

The visual shows the three types of consoles:

MCS MCS consoles (locally attached and SUBSYSTEM)—up to 99 defined through the CONSOLxx parmlib member.

EMCS Extended MCS (EMCS) consoles—programmable consoles defined and activated through an intended programming interface (the MCSOPER macro with REQUEST=ACTIVATE). The number of active EMCS consoles is not restricted.

System Console integration is supported by MVS and allows the hardware system console to support both hardware functions and the operating system IPL, recovery, and problem analysis.

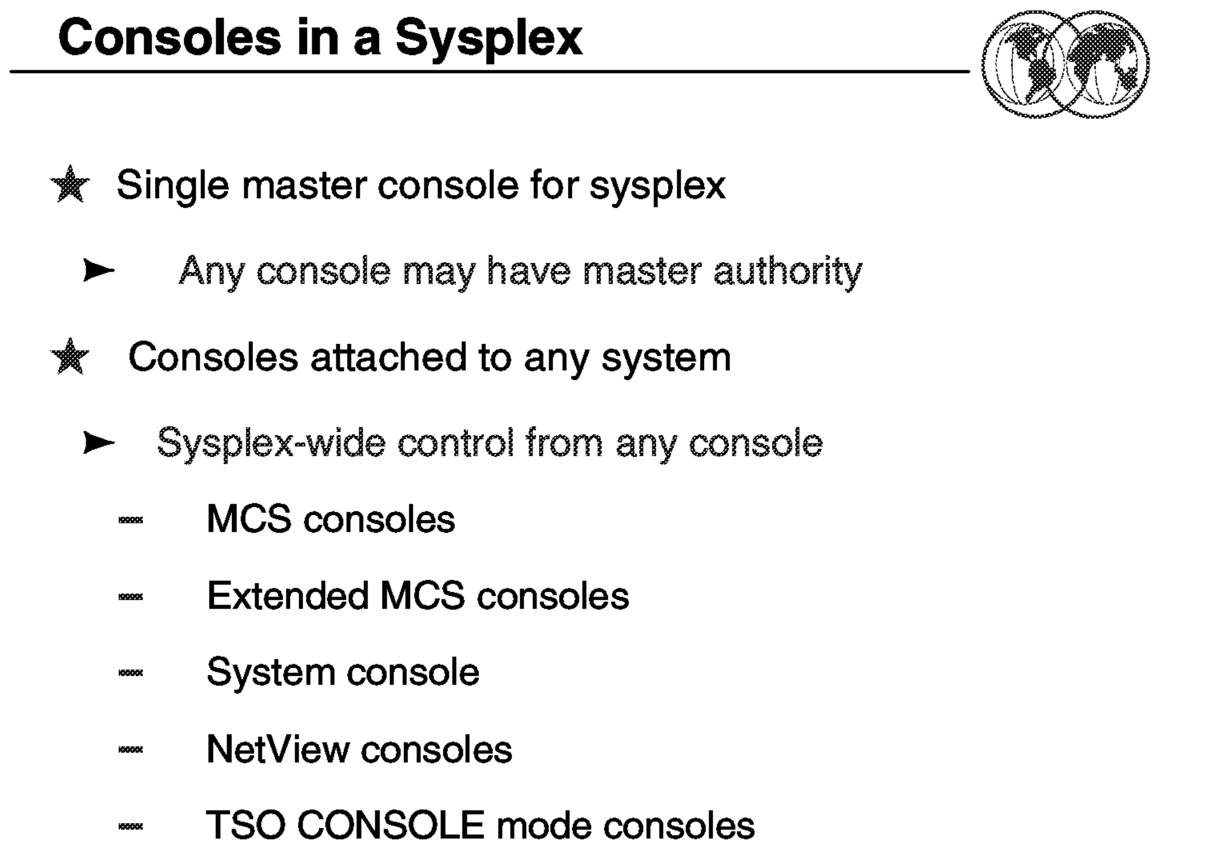
The system console in problem determination mode is automatically defined by MVS during system initialization.

In a sysplex, a console can be active on any system in a sysplex and can provide sysplex-wide control. MCS uses XCF services for command and message transportation between systems and thus provides a single system image for the operators. MCS multisystem support features:

- Sysplex-wide action message retention facility (ARMF)
- Sysplex-wide unique reply IDs
- Sysplex-wide command routing through:

- ROUTE operator command
- Command prefix facility (CPF)
- CMDSYS setting for a console (through the CONSOLE statement in the CONSOLxx parmlib member or CONTROL V,CMDSYS= operator command)

Normal message presentation is controlled by a console′s ROUTCDE and MSCOPE settings, and the UD (undeliverable message) attribute.



Consoles in a Sysplex

- Single master console for sysplex
  - Any console may have master authority
- Consoles attached to any system
  - Sysplex-wide control from any console
    - MCS consoles
    - Extended MCS consoles
    - System console
    - NetView consoles
    - TSO CONSOLE mode consoles

Figure 81. Consoles in a sysplex

4.2.1 Consoles in a sysplex

The introduction of a sysplex into the MVS environment provides a simpler and more flexible way to operate consoles in a multisystem environment. Many changes were introduced into multiple console support (MCS) to support the sysplex environment. These changes began with MVS/ESA Version 4 and have continued with each new MVS release.

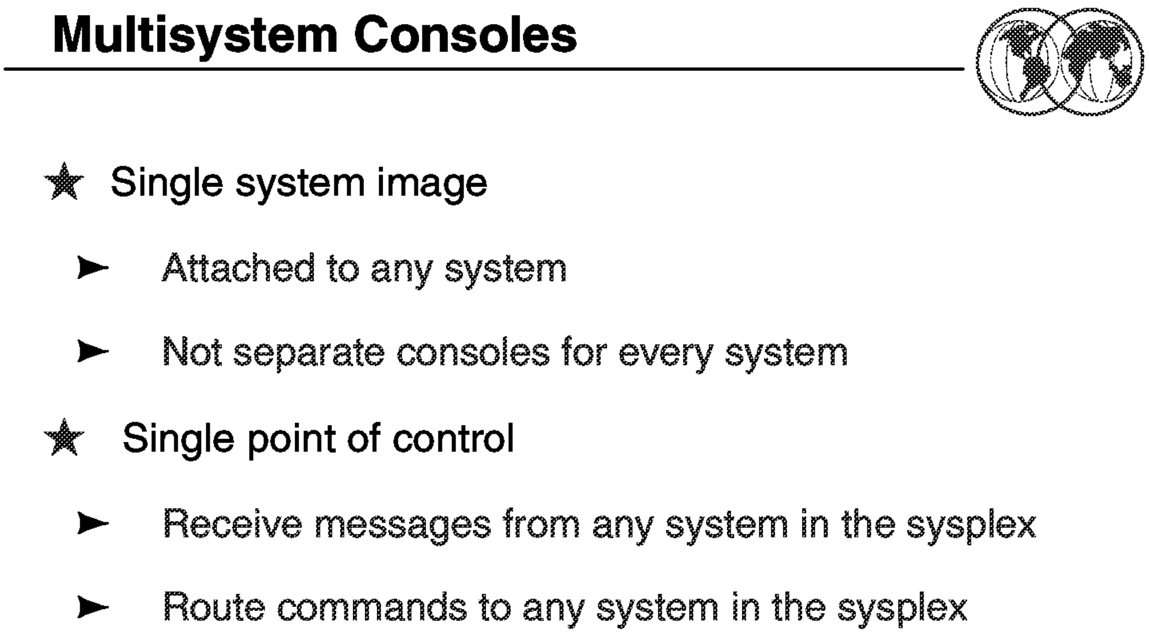
In a sysplex, MCS consoles can:

- Be attached to any system
- Receive messages from any system in the sysplex
- Route commands to any system in the sysplex

Therefore, new considerations need to be made when defining MCS consoles in this environment, such as:

- There is no requirement that each system have consoles attached.
- The 99 console limit for the sysplex can be extended with the use of extended MCS consoles. This adds greater flexibility when attaching consoles.
- A sysplex, which can have up to 32 systems, can be operated from a single console.
- Multiple consoles can have master command authority.

With MCS consoles in a sysplex, no matter where they are attached, it is possible to control any system in the sysplex. The ability to assign each console a unique name and unique characteristics greatly eases the task of system management.



Multisystem Consoles

- Single system image
  - Attached to any system
  - Not separate consoles for every system
- Single point of control
  - Receive messages from any system in the sysplex
  - Route commands to any system in the sysplex

Figure 82. Multisystem consoles

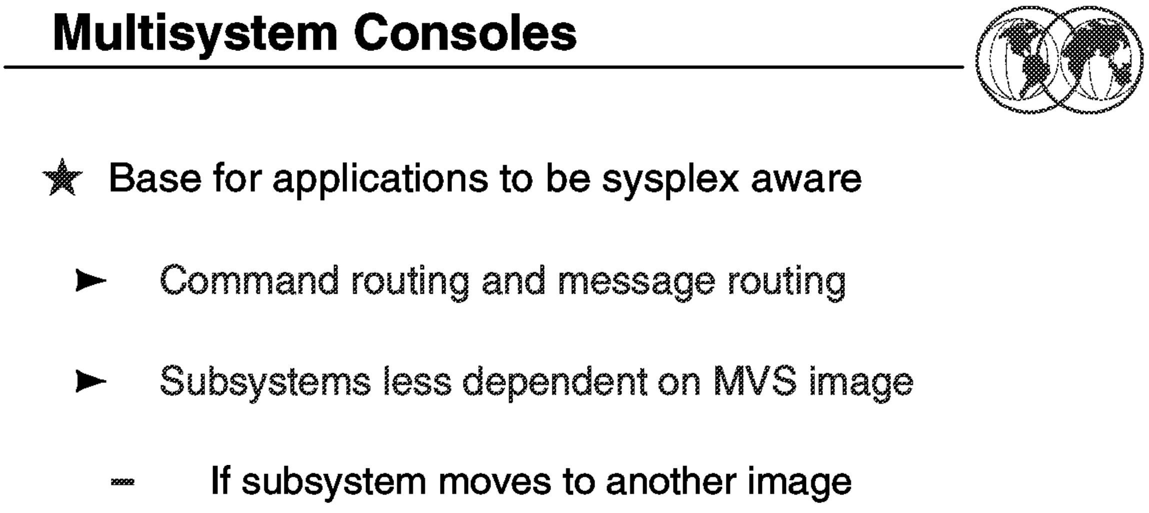
4.2.2 Multisystem consoles

The major goals of a sysplex from a systems management point of view are to provide:

- Single system image
- Single point of control
- Minimize human intervention

Multisystem console support has features to support single system image, single point of control, and minimal human intervention for those sysplex environments with these processors.

In a sysplex, the major tasks of operating an individual MVS image do not change very much. Consoles are used to receive messages and issue commands to accomplish system tasks. With the existence of multiple MVS images and multiple subsystems, the potential exists for the operator to receive an enormous number of messages, since there is the capability to route all messages from all MVS systems to a single console.



Multisystem Consoles

- Base for applications to be sysplex aware
  - Command routing and message routing
  - Subsystems less dependent on MVS image
    - If subsystem moves to another image

Figure 83. Multisystem consoles

4.2.3 Multisystem consoles

Multisystem consoles can be used by applications to be sysplex aware due to the fact that no matter on which system the applications exists, command routing and message routing can be done. The application does not have to exist on the same system as a console used to communicate with it.

Message Flow in a Sysplex

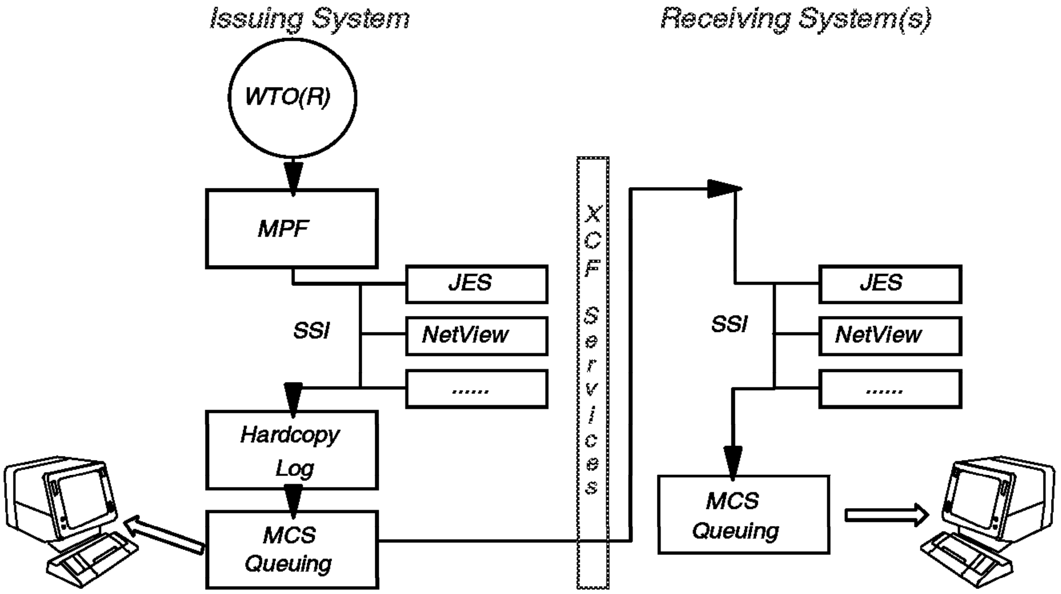


Figure 84. Message flow in a sysplex

4.2.4 Message flow in a sysplex

In a sysplex, a message is routed to all active consoles on all systems that are eligible to receive that particular message.

The visual shows the message flow in a sysplex. It is important to understand this flow because in a sysplex environment, message flow is changed to send messages issued on one system to other systems in the sysplex using XCF services.

WTO(R) The MVS write to operator (WTO) and the write to operator with reply (WTOR) macro services cause messages to be routed to the consoles and the hardcopy log.

MPF First on the issuing system, the message is processed by the message processing facility (MPF). This processing is based on entries in the MPFLSTxx parmlib member.

MPF processing allows an installation to influence how WTO and WTOR messages are to be processed. Through the MPFLSTxx member, you can specify some processing options for a message.

SSI Following MPF processing, the message is broadcast to all active subsystems that request to receive control for the WTO SSI function code 9. The subsystem must use the IEAVG700 interface to indicate that all WTO and WTORs are to be broadcast. The message is presented to each subsystem in turn. Each subsystem may inspect the message and process it as appropriate. A subsystem can alter write-to-operator queue element (WQE) fields, in which case later subsystems on the SSI will see the changed

WQE. A WQE is an internal control block that contains the message text and all related information for that message. The IHAWQE macro maps the WQE fields.

For example, when NetView is using the SSI rather than an extended MCS console for MVS communication, NetView on the SSI inspects all messages to see whether they are marked by MPF as eligible for automation. NetView intercepts automation message text and selected attributes from the WQE and sends the data to the NetView address space for further processing. NetView does not modify the actual WQE.

Log After the message has been inspected by all active subsystems, it is written to the hardcopy log (usually the SYSLOG data set, the operations log (OPERLOG), or both) unless hardcopy logging is suppressed by an exit. OPERLOG is a log stream maintained in a Coupling Facility that uses the system logger to record and merge communications about programs and system functions from each system in a sysplex. The messages are logged using message data blocks (MDBs), which provide more data than is recorded in the SYSLOG.

Queueing Finally, the message is routed for display on the appropriate MCS and extended MCS consoles. The routing may require message transportation using XCF services to other systems in the sysplex because some receiving consoles may not be physically attached to the system where the message was issued.

After the XCF transportation on the receiving system, the message goes through the SSI loop, but it is not logged, and finally the message is processed by the message queueing tasks to be displayed on the consoles.

If a message is destined for a specific console that is not active in the sysplex, it is logged and discarded unless it is an action message or WTOR message, in which case it is processed as an undelivered message. It is sent to all active consoles receiving UD messages. The master console is a UD receiver by default.

Messages that are already *delivered* to an active extended MCS console, but not yet *retrieved*, are purged from the MCS queues when the console is deactivated; that is, unprocessed queued messages are not rerouted.

The MSCOPE specification on the CONSOLE statement in the CONSOLxx parmlib member allows you to screen those systems in the sysplex from which a console is to receive messages not explicitly routed to the console.

Command Flow in a Sysplex

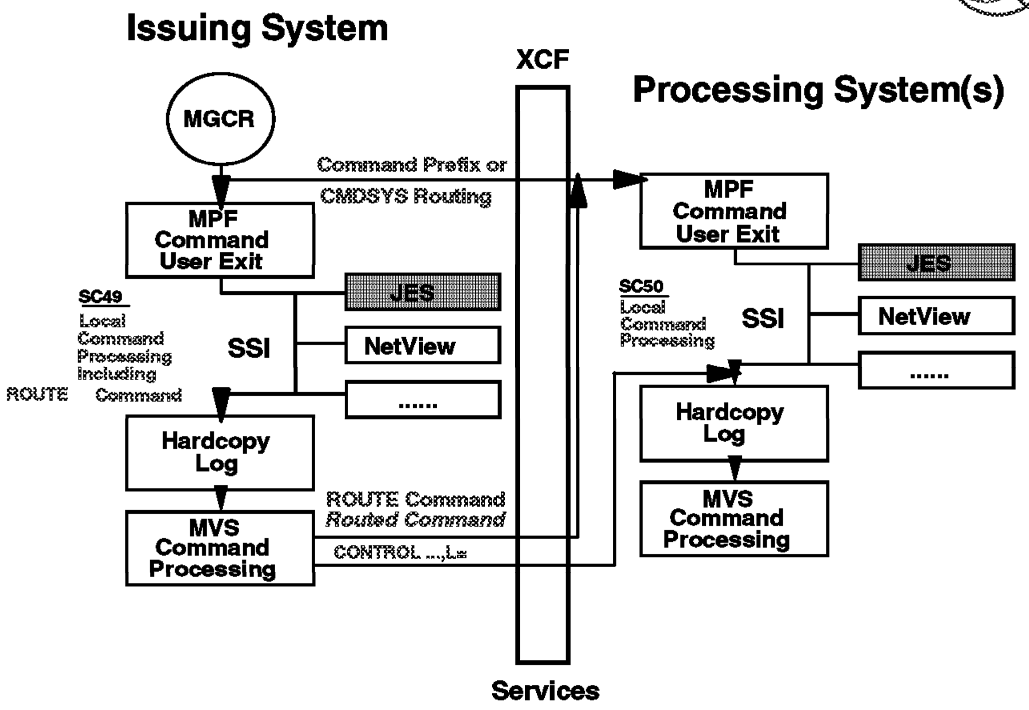


Figure 85. Command flow in a sysplex

4.2.5 Command flow in a sysplex

When an operator command is entered through the MGCR(E) macro service, the following processing flow takes place:

- *Command prefix and CMDSYS routing*

  If the command contains a prefix or the console has a CMDSYS specification that directs the command to another system, the command is immediately transmitted using XCF services to the processing system.

  Note: For command prefix and CMDSYS routing, the command is first transported to the receiving system before any system symbol substitution takes place.

  A REPLY command is sent to the system where the WTOR was issued. If the REPLY command text contains system symbols, substitution occurs on the receiving system.

- *MPF command user exit*

  If the command is not transmitted to another processing system, it is processed on the issuing system by the installation MPF command exit routines. The exits are specified using the .CMD statement in the MPFLSTxx parmlib member. These exits can perform authority checking, modify the command text, or the command processing.

  Note: For commands containing system symbols, substitution has occurred before the exit is entered.

- *SSI processing*

The command is then broadcast on the subsystem interface (SSI) to all active subsystems. Each subsystem inspects the command and decides whether to process it. The subsystems base the decision on the command prefix characters of the command string. For example, by default, NetView looks for a percent sign (%) and processes the commands starting with the % sign.

When a subsystem decides to process the command, the command is passed to subsystem processing, and a return code is set to indicate that the command was processed by a subsystem.

Note: At this point in processing, all system symbol substitution has occurred. The original command text is also available.

- *Hardcopy log*

  After the command has been examined by all active subsystems, it is logged to the hardcopy log (usually SYSLOG or OPERLOG).

  Note: The hardcopy log contains the command before any system symbols contained in the command have been substituted and, it also contains the command after substitution has occurred.

- *MVS command processing*

  If none of the subsystems has marked the command as having been processed, it is assumed to be an MVS command and is passed to the appropriate MVS command processor. If a command processor does not exist, an error message is issued stating that the command is invalid.

Console Parmlib Members

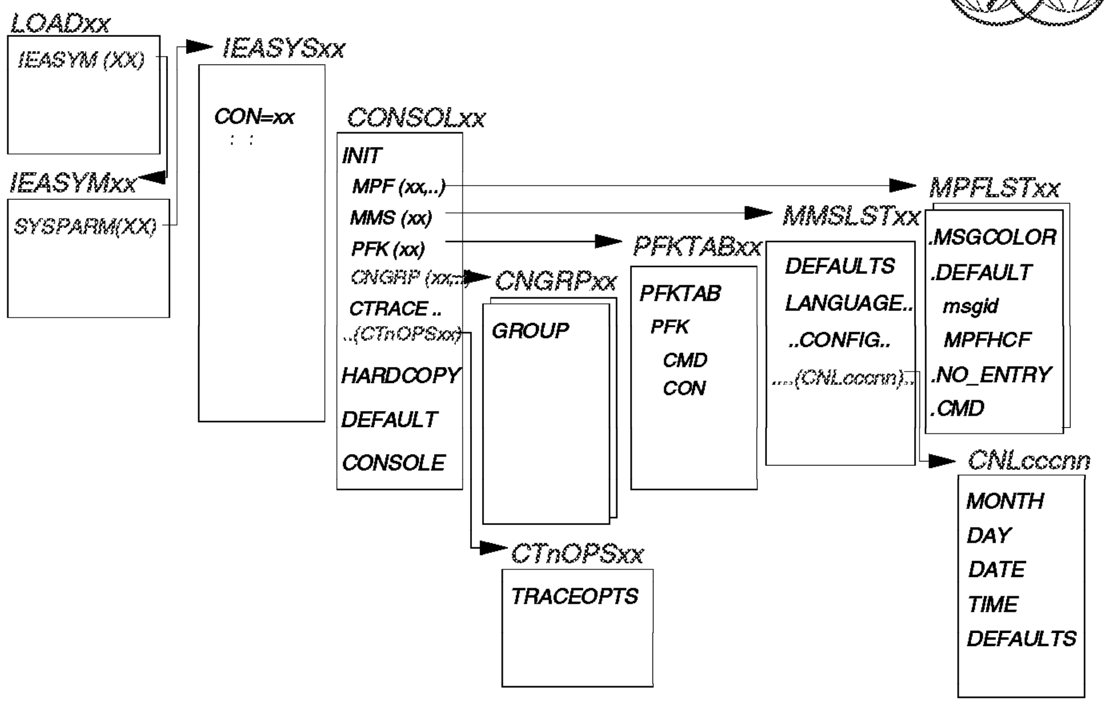


Figure 86. Console parmlib members

4.2.6 Console parmlib members

The visual summarizes the console-related parmlib members, their *chaining*, and also lists the various statements in each member.

Note: You must be at the MVS/ESA SP Version 5.2 level for the LOADxx to point to the IEASYMxx.

Besides the CONSOLxx parmlib member, several other related parmlib members affect console operations and are shown in the visual. Within the CONSOLxx member, you may reference:

CNGRPxx The CNGRPxx parmlib member defines console groups for alternate console switching.

CTnOPSxx The CTnOPSxx parmlib member specifies the component tracing options for the operations services (OPS) component.

MMSLSTxx The MMSLSTxx parmlib member is used to display translated U.S. English messages into another language that your installation has provided.

The MMSLSTxx parmlib references the message configuration member CNLcccxx parmlib member which defines how translated messages are to be displayed at your installation. In a message configuration member, you specify the time and date format for translated messages using the MONTH, DAY, DATE, and TIME statements.

The primary objective of the MVS message service is to provide a programmable interface for message translation. The MVS message service allows components of, or products running on, MVS to translate U.S. English messages into the user′s native language.

Note: MCS messages displayed on locally attached MCS consoles are not translated. When the *MVS message services* (MMS) is active, the TSO/E CONSOLE command attempts to translate console messages to the primary language specified in the user′s profile (UPT).

MPFLSTxx The MPFLSTxx parmlib member is used for message processing control.

PFKTABxx The PFKTABxx parmlib member is used to define any PFK tables for the MCS consoles.

CNLcccxx The CNLcccxx parmlib member is used to define how translated messages are to be displayed.

LOADxx Parmlib Member

In order to use a different SYSNAME in IEASYSxx before MVS/ESA 5.2, multiple LOADxx members had to exist in SYSx.IPLPARM or SYS1.PARMLIB to IPL multiple images that pointed to different IEASYSxx members via the SYSPARM statement. With MVS/ESA 5.2 and all following systems, it is now possible to specify one LOADxx member pointing to a new parmlib member (IEASYMxx).

Chapter 5. Automatic restart management

Automatic restart management is key to automating the restarting of subsystems and applications (referred to collectively as applications) so they can recover work they were doing at the time of an application or system failure and release resources, such as locks, that they were holding. With automatic restart management policy, you can optionally control the way restarts are done. You can use the policy defaults that IBM provides, or you can override them.

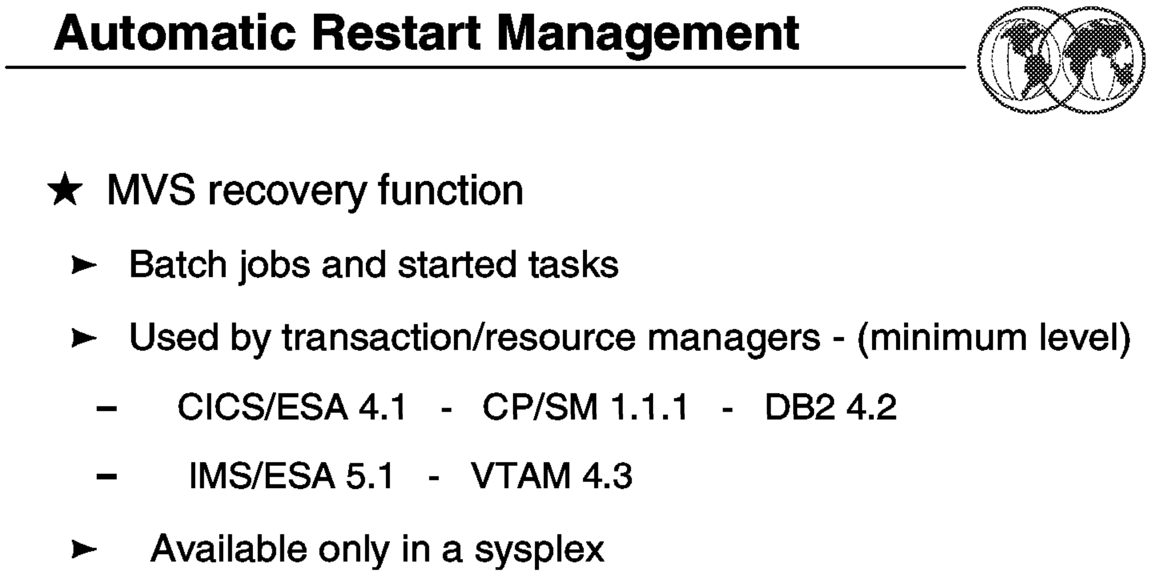
In a sysplex environment, a program can enhance its recovery potential by registering as an element of automatic restart management. Automatic restart management can reduce the impact of an unexpected error to an element because MVS can restart it automatically, without operator intervention. In general, MVS restarts an element when:

- The element itself fails. In this case, MVS restarts the element on the same system.
- The system on which the element was running unexpectedly fails or leaves the sysplex. In this case, MVS restarts the element on another system in the sysplex; this is called a cross-system restart.

In general, your installation may use automatic restart management in two ways:

- To control the restarts of applications (such as CICS) that already use automatic restart management as part of their recovery.
- To write or modify installation applications to use automatic restart management as part of recovery.

During planning before installation of the MVS systems in the sysplex, you can define a sysplex failure management (SFM) policy, and an automatic restart management policy. The goals of SFM and automatic restart management are complementary. SFM keeps the sysplex up and running; automatic restart management keeps specific work (batch jobs or started tasks) in the sysplex up and running. SFM manages system-level problems and partitions systems out of the sysplex in response to those problems. Automatic restart management manages subsystems and restarts them appropriately after either subsystem-level or system-level failures occur.



Automatic Restart Management

★ MVS recovery function

- ➤ Batch jobs and started tasks
- ➤ Used by transaction/resource managers - (minimum level)
  - – CICS/ESA 4.1 - CP/SM 1.1.1 - DB2 4.2
  - – IMS/ESA 5.1 - VTAM 4.3
- ➤ Available only in a sysplex

Figure 87. Automatic restart management

5.1 Automatic restart management

Automatic restart management is an MVS recovery function that can improve the availability of specific batch jobs or started tasks. When a job or task fails, or the system on which it is running fails, automatic restart management can restart the job or task without operator intervention.

The goals of SFM and automatic restart management are complementary. While SFM keeps the sysplex running, automatic restart management keeps specific work in the sysplex running. If a job or task fails, automatic restart management restarts it on the same system it was running on at the time the failure. If a system fails, automatic restart management restarts the work on other systems in the sysplex; this is called a cross-system restart.

A program cannot use both automatic restart management and checkpoint/restart. If a program using checkpoint/restart tries to register with automatic restart management, the request is rejected.

The purpose of automatic restart management (ARM) is to provide fast, efficient restarts for critical applications when they fail. ARM is an MVS recovery function that improves the time required to restart an application by automatically restarting the batch job or started task (STC) when it unexpectedly terminates. These unexpected outages may be the result of an abend, system failure, or the removal of a system from the sysplex.

A primary availability requirement for all system environments is to reduce the duration and impact of an outage. The sysplex environment helps to achieve this objective through the following recovery strategies:

- Detection and isolation of failures - detect a failure and route work around it.
- Recovery of resources - release critical resources as soon as possible so that other programs and systems can acquire them. This reduces contention and speeds up the recovery or restart of work affected by the failure.
- Recover lost capacity - restart or resume processing that was affected by the failure. For example, in a database environment, the DB manager should be restarted quickly so that its log recovery can be processed.

The intended exploiters of the ARM function are the jobs and STCs of certain strategic transaction and resource managers. Some of these include:

- CICS/ESA
- CP/SM
- DB2
- IMS/TM
- IMS/DBCTL
- ACF/VTAM

In general, an installation may use automatic restart management in two ways:

- To control the restarts of applications (such as CICS) that already use automatic restart management as part of their recovery.
- To write or modify installation applications to use automatic restart management as part of recovery.

For information on automatic restart management, see *OS/390 MVS Setting Up a Sysplex*, GC28-1779, *OS/390 MVS Sysplex Services Guide*, GC28-1771, and *OS/390 Parallel Sysplex Systems Management*, GC28-1861.

ARM Environment

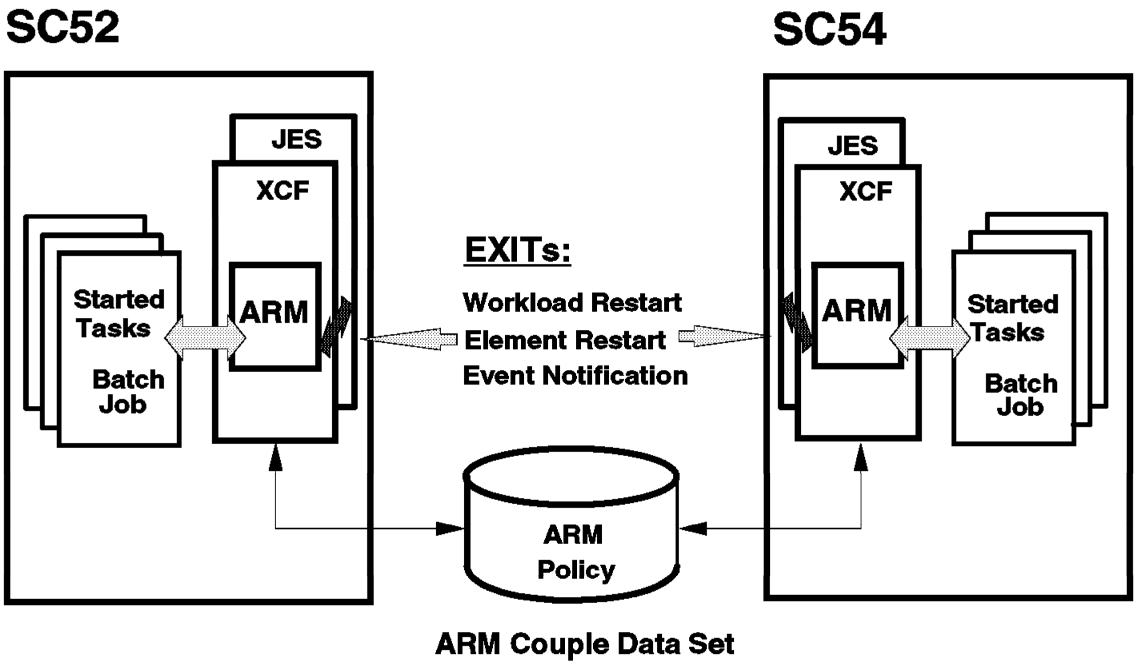


Figure 88. ARM environment

5.1.1 ARM environment

Automatic restart management is a function introduced in MVS/ESA SP 5.2.0. It runs in the cross-system Coupling Facility (XCF) address space and maintains its own dataspaces. ARM requires a primary couple data set to contain policy information as well as status information for registered elements. It supports both JES2 and JES3 environments.

In addition to the policy, three exit points are provided where subsystem and installation-written exit routines can be invoked to cancel a restart or influence how it is done. This gives you extended control and can simplify or enhance the policy that you build and activate. The exits are:

- The workload restart exit (IXC_WORK_RESTART)
- The element restart exit (IXC_ELEM_RESTART)
- The event exit

In a sysplex with a mixture of systems, some at lower release levels, ARM is operational *only* on those systems that have MVS/ESA SP 5.2.0 and JES 5.2, and on all subsequent releases of OS/390.

A system must be connected to an ARM couple data set with an active automatic restart management policy.

Create ARM Couple Data Set

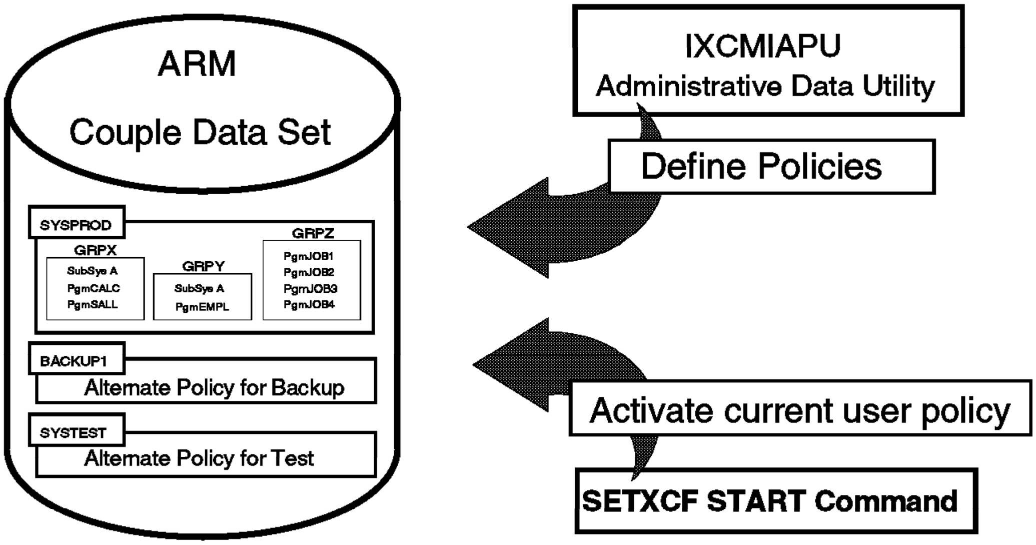


Figure 89. Create ARM couple data set and policy

5.1.2 Create ARM couple data set and policy

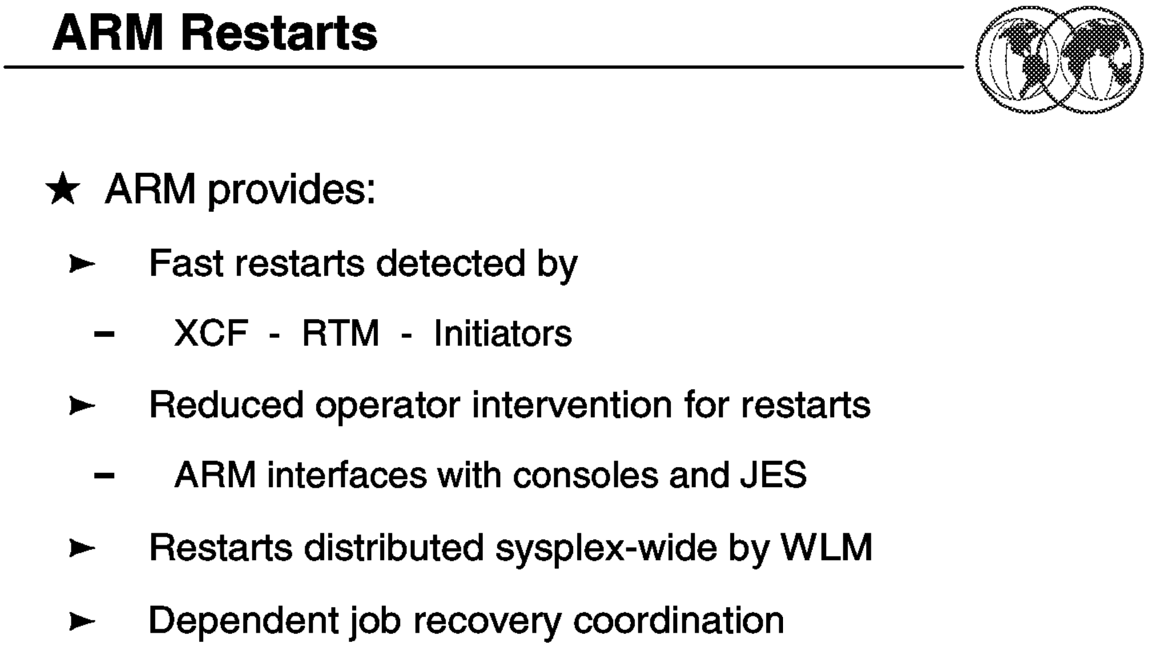
The visual shows the key activities necessary to create an ARM couple data set and an ARM policy:

- Create a primary and alternate couple data set:
  - Create and submit a ARM couple data set formatting job (IXCL1DSU).
- Add the primary couple data set to the sysplex:
  - Use the command `SETXCF COUPLE,TYPE=ARM,PCOUPLE=(SYS1.XCF.ARM10)` to add the primary ARM data set and make your systems ARM capable.
- Add the alternate couple data set to the sysplex
  - Use the command `SETXCF COUPLE,TYPE=ARM,ACOUPLE=(SYS1.XCF.ARM20)` to add the secondary ARM data set.

Define an ARM policy:

- If started tasks or jobs need a special restarting policy, then an ARM policy should be defined.
- Start the defined policy:
  - The command `SETXCF START,POLICY,TYPE=ARM,POLNAME=name` starts the defined policy.

  To provide program recovery through automatic restart management, an installation has to activate a policy through the `SETXCF START` command—this can be an installation-written policy or the IBM-supplied policy defaults. Because an installation-written policy may have an effect on your program, you need to understand how your installation uses automatic restart management for recovery in a sysplex, and code the program with these factors in mind.



ARM Restarts

★ ARM provides:

- ➤ Fast restarts detected by
  - – XCF - RTM - Initiators
- ➤ Reduced operator intervention for restarts
  - – ARM interfaces with consoles and JES
- ➤ Restarts distributed sysplex-wide by WLM
- ➤ Dependent job recovery coordination

Figure 90. ARM restarts

5.1.3 ARM restarts

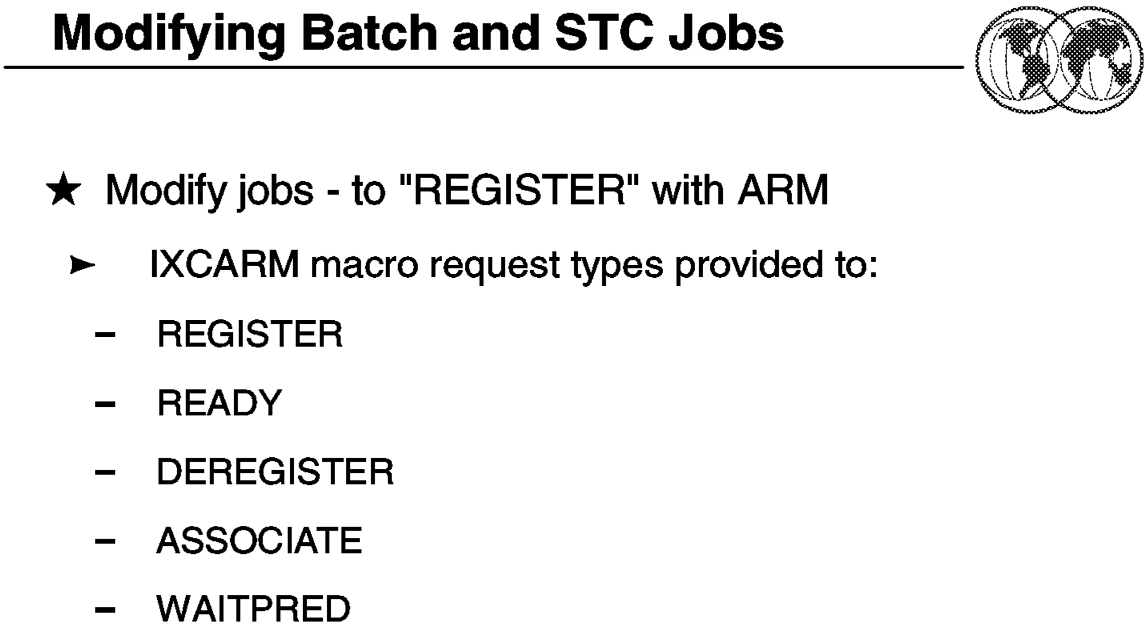
The purpose of automatic restart management (ARM) is to provide fast, efficient restarts for critical applications when they fail. ARM is an MVS recovery function that improves the time required to restart an application by automatically restarting the batch job or started task (STC) when it unexpectedly terminates. These unexpected outages may be the result of an abend, system failure, or the removal of a system from the sysplex.

The need for restarts is detected by either XCF, recovery termination management (RTM), or the initiator.

ARM interfaces with MCS console support and JES to provide its recovery mechanisms.

Workload Manager (WLM) provides statistics on the remaining processing capacity in the sysplex.

ARM minimizes failure impact by doing fast restarts without operator intervention in the process. Related groups of programs can be restarted, and if programs are start-order dependent, it assures that the order dependency is honored. The chance of work being restarted on systems lacking the necessary capacity is reduced as the best system is chosen, based on the statistics returned to ARM by the Workload Manager.



Modifying Batch and STC Jobs

- ★ Modify jobs - to "REGISTER" with ARM
  - ► IXCARM macro request types provided to:
    - – REGISTER
    - – READY
    - – DEREGISTER
    - – ASSOCIATE
    - – WAITPRED

Figure 91. Modifying batch and STC jobs

5.1.4 Modifying batch and STC jobs

To use ARM, jobs must be modified to register with ARM using the IXCARM macro services. In a sysplex environment, a program can enhance its recovery potential by registering as an *element* of automatic restart management. Automatic restart management can reduce the impact of an unexpected error to an element because MVS can restart it automatically, without operator intervention.

The automatic restart management service routine is given control from the IXCARM macro and is used to:

REGISTER Register a user of automatic restart management services.

READY Mark a user of automatic restart management services as ready to accept work.

DEREGISTER Deregister a user of automatic restart management services.

ASSOCIATE Associate a user of automatic restart management services with another user for takeover or restart purposes. This allows a program to identify itself as the backup program for another element of the automatic restart manager. This identification tells MVS that the other element should not be restarted unless this backup program is deregistered.

WAITPRED Wait until predecessor elements have been restarted if applicable. This indicates that MVS should delay the restart for this program until MVS completes the restart of a related program, which is called a predecessor element.

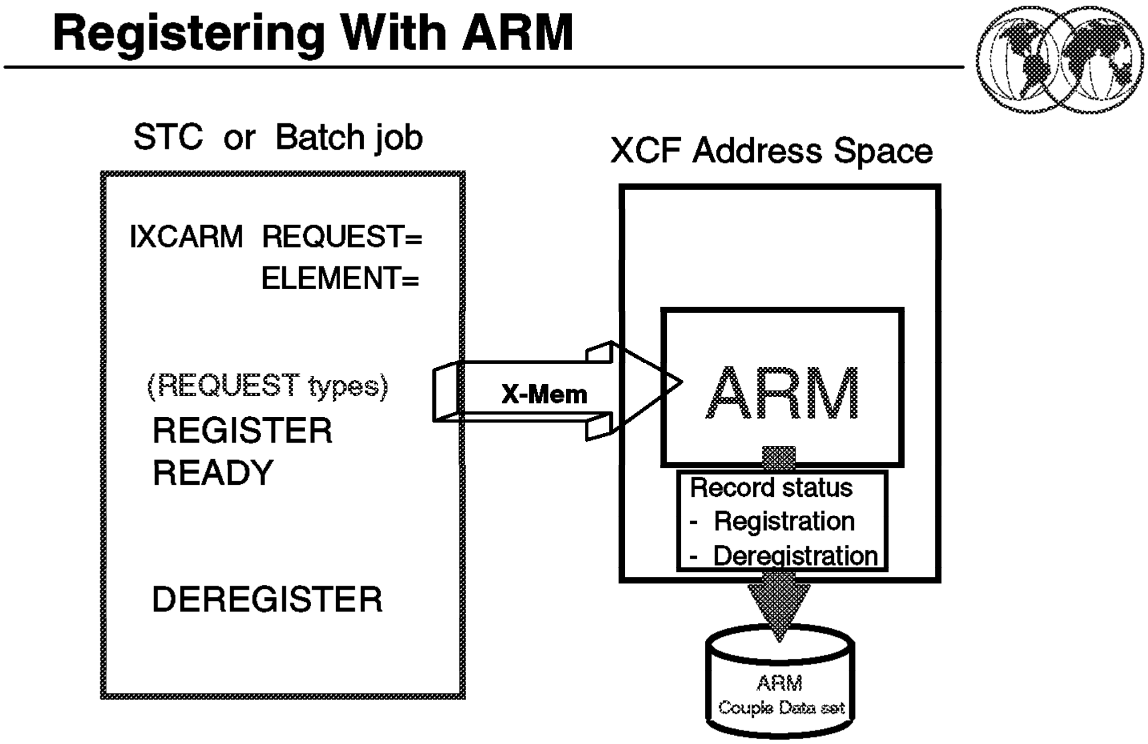


Figure 92. Registering with ARM

5.1.5 Registering with ARM

For a batch job or started task to be ARM restartable, they must register with ARM using an authorized macro service IXCARM. The registration requires the specification of an *element* name. This element name must be unique in the sysplex as that is how ARM identifies jobs under its control for restart. The element name can be from one to sixteen characters. The three basic ARM services that a job must use as shown in the visual are:

REGISTER Early in the initialization processing for a job, a program that wants to use ARM for restart must issue the IXCARM REQUEST=REGISTER macro in order to register. Also specified on the macro request is the element name.

Optionally, you can specify restart parameters and an event exit.

After the job or task has issued the IXCARM macro with the REGISTER parameter, MVS can automatically restart the program when an unexpected failure occurs. You can specify restart parameters on the REGISTER request; however, restart parameters in an installation-written policy override parameter values specified on the IXCARM macro.

READY When a program that has issued the REGISTER request has completed initialization and is ready to process, it must issue the IXCARM REQUEST=READY request. This indicates that the program is ready to do work.

DEREGISTER Before a job completes its normal termination processing, the program must issue an IXCARM REQUEST=DEREGISTER request to remove the element and all ARM restart possibilities.

You can deregister from automatic restart management when the job or task no longer needs to be restarted. If a program fails after it deregisters, MVS will not restart the program.

ARM Element States

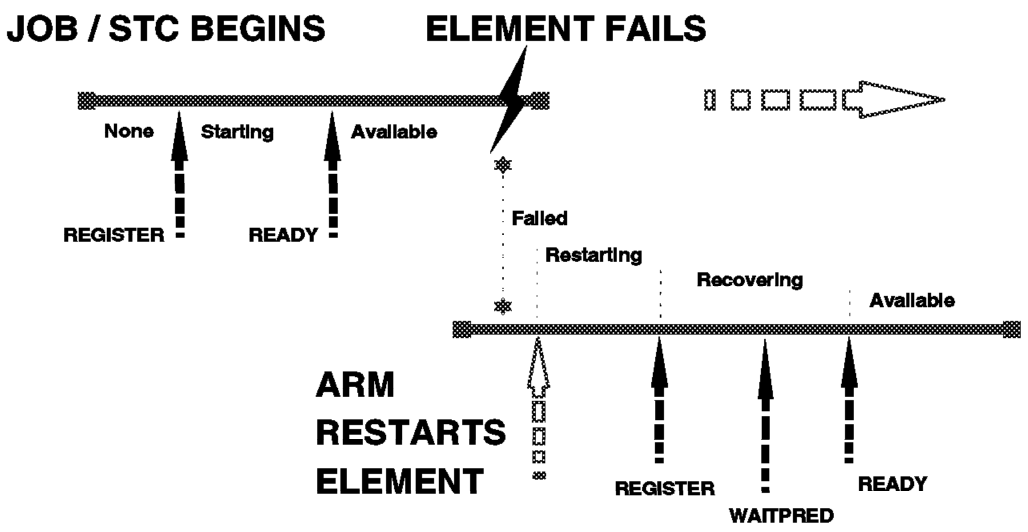


Figure 93. ARM element states

5.1.6 ARM element states

The visual shows an element′s use of ARM services and how they affect the state of that element.

ARM puts each element into one of five states to identify the last interaction that the element had with it. These states are:

Starting From the initial registration (IXCARM-Register) of a program as an ARM element to it subsequently becoming ready (IXCARM-Ready).

Available From the time the element becomes ready (IXCARM-Ready) until the element either deregisters from ARM or terminates.

Failed From when ARM detects the termination of an element, or termination of the system on which the element had been running, until ARM initiates a restart of the element.

Restarting From ARM′s initiation of a restart until the subsequent reregistration of the element.

Recovering From the element′s registration after an ARM restart to its subsequent issuing of IXCARM-Ready.

Initially, an element is unknown to ARM. The first interaction with ARM occurs when the program makes itself known to ARM by invoking the register service and specifying the element name under which it is to be registered. ARM sets the state of the element to *starting*.

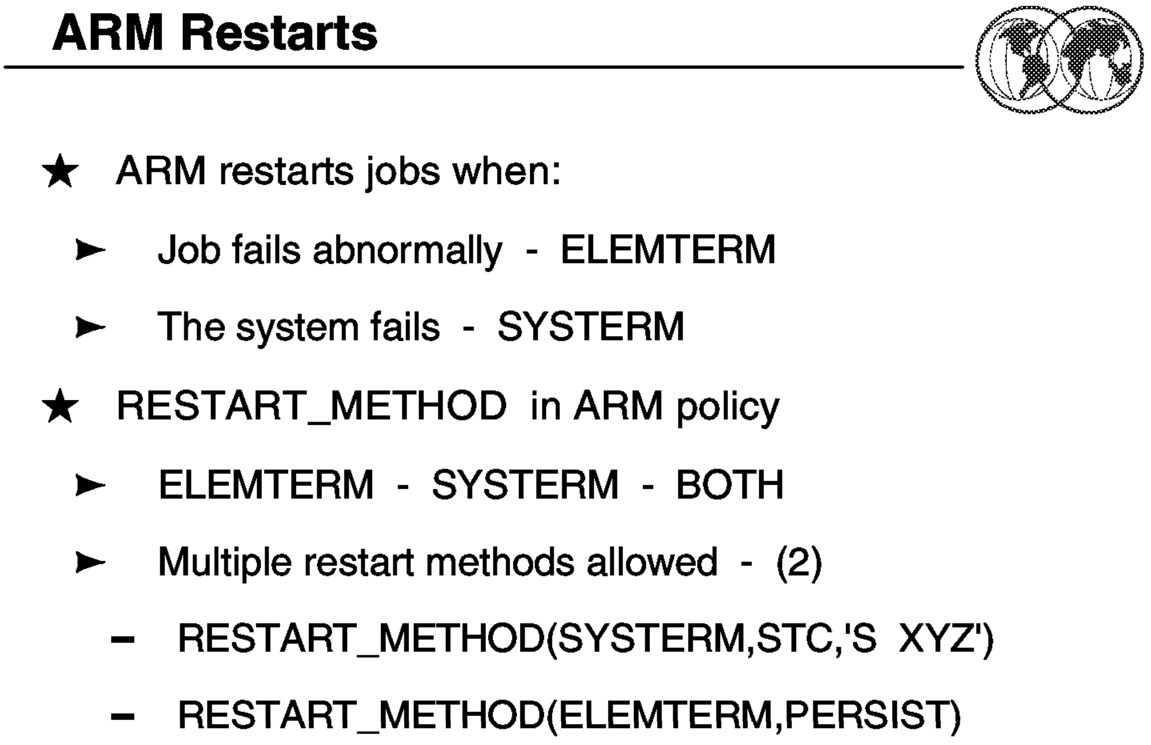
When the element is ready to accept new work, it invokes the ready service, and ARM sets the state of the element to *available*. Once the element′s state is available, there are no more interactions with

ARM until the element terminates. Before terminating normally, the element invokes the deregister service, which again makes the element unknown to ARM.

When ARM detects that the element has unexpectedly terminated, that is, the program has terminated without invoking the deregister service, or that the system on which the element had been running has left the sysplex, ARM sets the element′s state to *failed* as part of its processing to restart the element. An element will be in the failed state for a very brief interval.

When ARM restarts the element, it sets the state to *restarting*. When a restarting element subsequently invokes the register service, ARM sets the state to *recovering*. Once the element is prepared to accept new work, it notifies ARM by invoking the ready service, which causes ARM to set the element′s state to *available*.

The state of a given element will be part of the information provided when ARM status is requested through the DISPLAY XCF command for one or more elements.



ARM Restarts

- ★ ARM restarts jobs when:
  - ➤ Job fails abnormally - ELEMTERM
  - ➤ The system fails - SYSTERM
- ★ RESTART_METHOD in ARM policy
  - ➤ ELEMTERM - SYSTERM - BOTH
  - ➤ Multiple restart methods allowed - (2)
    - – RESTART_METHOD(SYSTERM,STC,'S XYZ')
    - – RESTART_METHOD(ELEMTERM,PERSIST)

Figure 94. ARM restarts

5.1.7 ARM restarts

ARM has two restart types:

ELEMTERM Element termination

When an element unexpectedly fails, automatic restart management is given control during end-of-job and end-of-memory termination after all other recovery has taken place. If the job or task terminating is an element of the automatic restart manager and should be restarted, then MVS:

1. Gives control to the element-restart exit
2. Gives control to the event exit
3. Restarts the element
4. Issues an ENF signal when the element re-registers with the automatic restart manager

SYSTERM System termination (SYSTERM)

When a system unexpectedly fails, automatic restart management determines if any elements were running on the system that failed. If those elements can be restarted on another system, and cross-system restarts are allowed, MVS does the following for each system on which the elements will be restarted:

1. Gives control to the workload restart exit
2. For each element that will be restarted:
   a. Gives control to the element restart exit

b. Gives control to the event exit
c. Restarts the element
d. Issues an ENF signal when the element re-registers with the automatic restart manager

5.1.7.1 RESTART_METHOD in ARM policy

The ARM policy contains entries for elements that determine what type of restart the element is to receive. In the ARM policy, RESTART_METHOD is optional. The default is RESTART_METHOD(BOTH,PERSIST). For started tasks only, the IXCARM macro parameter that overrides the default specification is STARTTXT.

The RESTART_METHOD has three options:

ELEMTERM Indicates that the persistent JCL or command text is to be overridden by the JCL data set or the command text specified in restart-type only when the element itself terminates.

SYSTERM Indicates that the persistent JCL or command text is to be overridden by the JCL data set or the command text specified in restart-type only when the system the element was running on terminates.

BOTH Indicates that the persistent JCL or command text is to be overridden by the JCL data set or the command text specified in restart-type when either the system the element was running on terminates, or when the element itself terminates.

Following are the three options, the restart type, for how to restart a failed element, as shown on the visual:

STC,′command-text′ Indicates that automatic restart management is to restart this element by issuing the command provided in ′command-text′.

PERSIST Indicates that MVS is to use the JCL or command text that previously started the element.

JOB,′jcl-source′ Indicates that MVS is to restart the element as a batch job. ′jcl-source′ is the name of the data set that contains the JCL for restarting the element. This data set name must be enclosed within single quotes. The data set must have the same data set characteristics (for instance, LRECL) as standard procedure libraries.

When ARM restarts an element that terminated with an ELEMTERM, this element is restarted on the same system where it was previously active. Even if the initiators of the system all are busy or stopped, JES tries to restart the job on that system. This is done by setting the system affinity of the restarting job to the same system.

A job that is being ARM-restarted when it fails with an ELEMTERM termination can never execute on a different system. This is because the job would try to register with ARM using the same element name on a different system than the restart system. This is not allowed, and the following return code and reason code are issued by ARM: (RC0008, RSN0150 of IXCARM REQUEST=REGISTER).

Restart on Same System

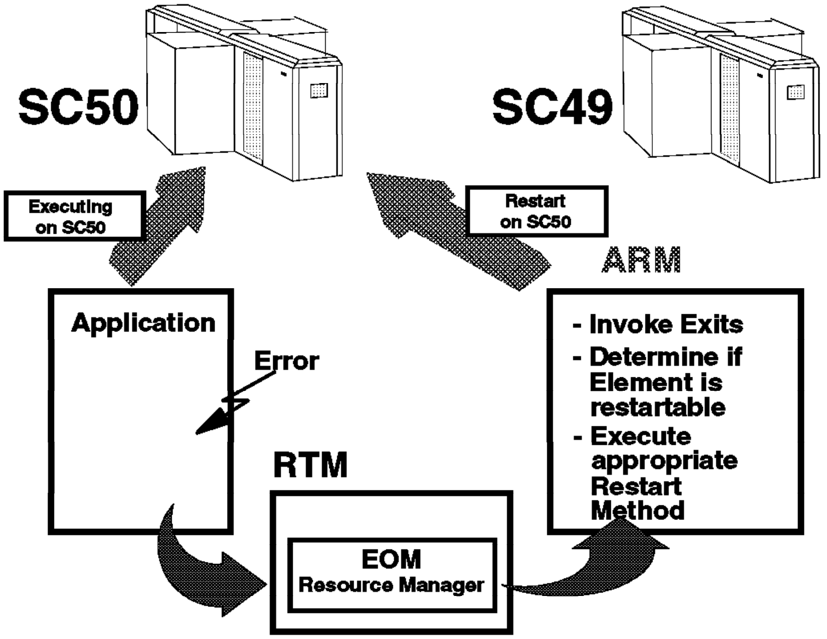


Figure 95. Restart on same system

5.1.8 Restart on same system

When an element ABENDS or otherwise terminates while registered with ARM, ARM restarts that element on the same system on which it had been running. Such ARM restarts are referred to as *restarts in place.* You should be aware that ARM restart processing does not include any kind of cleanup or recovery. That is considered internal to the element being restarted. Element related processing, such as, the releasing of *local* resources or the analysis of the impact of the termination to on-going work, is the responsibility of recovery routines of the element. The visual shows the various factors involved in restarting work on the same system.

ARM′s end-of-memory (EOM) resource manager determines if a terminating address space is a registered element. If so, it schedules the restarting of that element. The elements restarted from ARM′s EOM resource manager are started tasks. Restarts for batch jobs are performed in essentially the same way from the ARM end-of-job (EOJ) resource manager.

If the element to be restarted specified an event exit routine when it registered as an ARM element, then ARM calls the specified exit routine before restarting that element. This processing could instruct ARM *not* to restart the element. If there are any element restart exits defined, ARM also invokes them. An element restart exit routine can cancel the restart of a given element or change the method by which ARM restarts it.

The method in which ARM restarts an element depends on whether the element is a started task or a batch job. You can specify the restart method through either the ARM policy or an exit routine. A registering program can also specify a command (through the REGISTER macro) to be used for ARM restarts.

If an installation policy or an exit routine provides an override JCL data set or command text to restart an element, ARM determines if the data set name or command text contains any symbolic substitution parameters. If it does, these parameters are resolved before the command is issued or the JCL submitted. This resolution of symbolic substitution parameters is done using the values from the system on which the element initially registered. For restarts in place, ARM′s replacement of these values provides the same result as would occur if the normal MVS resolution had been done.

After initiating the restart, ARM loses contact with the element and cannot track how the restart is processing. ARM establishes a time interval to wait for the subsequent registration of the element from its initialization process. If the time interval expires before registration is received, then a *restart timeout* of the element is recognized. An error message is produced, an ENF is issued indicating that ARM is no longer handling restarts for the element, and all resources related to the element are cleaned up. Lastly, ARM deregisters the element. If the element was just delayed and registers after the restart timeout, then ARM will treat this as a new registration.

Once registered, the timeout interval is set for the element to become ready. If this expires, then a warning message is produced, but no other actions occur. The element would remain in the RECOVERING state. Finally, when the element requests the READY service, the state is set to AVAILABLE, and the restart processing is considered complete.

Restart on Different Systems

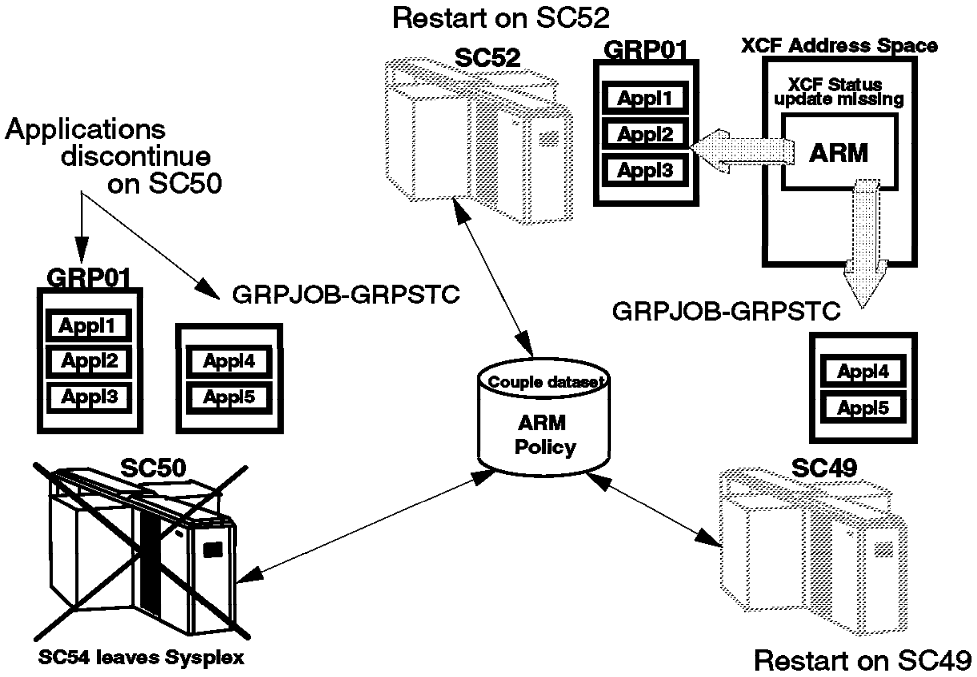


Figure 96. Restart on different systems

5.1.9 Restart on different systems

When a system fails or otherwise leaves the sysplex, ARM has the overall responsibility of restarting registered elements that had been running on the failed system, on the remaining systems in the sysplex. ARM depends on several other components to achieve this goal.

The visual shows the various factors involved in restarting work on the same system.

The first action taken is the detection and the initial termination of the failed system. Here is a summary of the actions performed by other components:

1. The detecting MVS XCF issues a *status update missing event* because the XCF failure detection interval has elapsed.
2. The XCF indeterminate status interval expires. Available options are:
   - Operator prompt
   - Reset after isolation
   - LPAR system reset or deactivation via ARF (PR/SM only)
3. The failed system is removed from the sysplex (sysplex partitioning):
   - Logical partition deactivation and acquiring storage resources via ARF (PR/SM only)
   - Multisystem XCF exploiters recover resources owned by failed system:
     - Console services - release console resources owned by the failed system
     - GRS - release global ENQs owned by the failed system

All remaining systems are notified of the system′s termination through the XCF group exit. XCF, on each MVS image in the sysplex, directly notifies ARM of the system′s termination. The ARM processing that initiates restart processing runs on the MVS image that first detected the system termination. The ARM processing on this system has several responsibilities:

1. Determine if ARM′s threshold for system terminations prohibits its restarting of the element that had been running on the failed system
2. Determine if any registered elements were executing on the failed system
3. Determine which elements can be restarted on another system
4. Determine which elements must be restarted on the same system (same restart group)
5. Determine the system on which to restart each restart group

When a system leaves the sysplex, ARM determines if there have been two or more other *SYSGONE* events within the last ten minutes. If so, ARM does not restart any of the elements from the system that experienced this latest SYSGONE event. ARM issues a message for this event. This is done in case there is some problem in ARM or in its elements that causes the system failure. If left unchecked, such a problem could cause ARM restarts to cascade the failure to many or all systems in the sysplex.

ARM attempts to restart the elements from a system that left the sysplex on the remaining systems that have the most available capacity. The steps that ARM follows to identify and prioritize the candidate systems are:

- Determine the relative available capacity and available CSA/ECSA of MVS/ESA SP 5.2.0 systems in WLM′s goal mode in the sysplex. ARM builds a list of these systems, sorted according to the available capacity. Any active MVS/ESA SP 5.2.0 system in WLM′s compatibility mode is then added to the bottom of this list.

- For each restart group that is to be restarted, ARM will then:

  1. Identify the systems that are in the same JES XCF group as the system that has terminated. Systems not in the same JES XCF group are deleted from the list of systems just created.

  2. Determine if the installation′s policy specifies either TARGET_SYSTEM or FREE_CSA for the restart group, and if so, eliminate from the list any system that does not meet the criteria defined by these parameters.

  The top system on the list is then assigned to restart this restart group. If there are no candidate systems left after these checks, ARM aborts the restart of that restart group.

On each system where restarts are to occur, ARM calls your workload restart exit if it is installed. This allows the installation to do things like cancel work that is presently running on that system to prepare for the elements that ARM is about to restart. ARM then initiates restart processing for the elements within the restart group. They are essentially started at the same time to allow for maximum parallelism during restart processing.

As described in the policy, there is a notion of restart levels among the elements. One or more elements may be given a level. This allows for sequencing of restart logic between elements. All elements in a particular level must become *ready* before ARM allows elements of a higher level in the same restart group to become ready.

Group Restart Overview

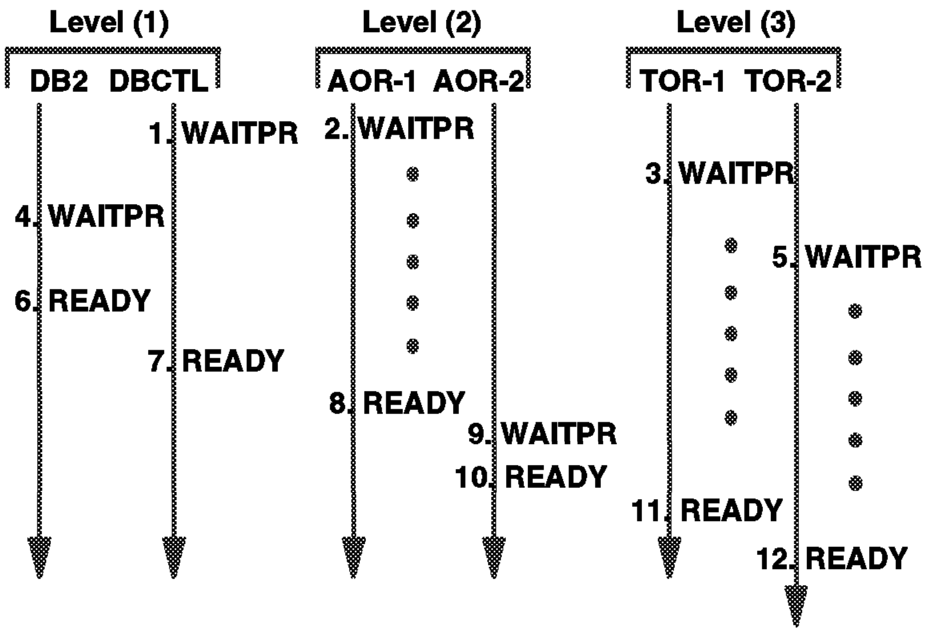


Figure 97. Group restart overview

5.1.10 Group restart overview

This example of a group restart shows how DB2 and CICS coordinate their restarts. Both DB2 and CICS can be started at the same time, including AORs and TORs. Since they are at different levels, as defined in the ARM policy, they must wait for higher levels to complete their initialization before they can continue, so they issue a WAITPRED to wait for a higher level to issue a READY.

When the ready status of a higher level is given, then the next level can continue its initialization.

ARM Exit Facilities

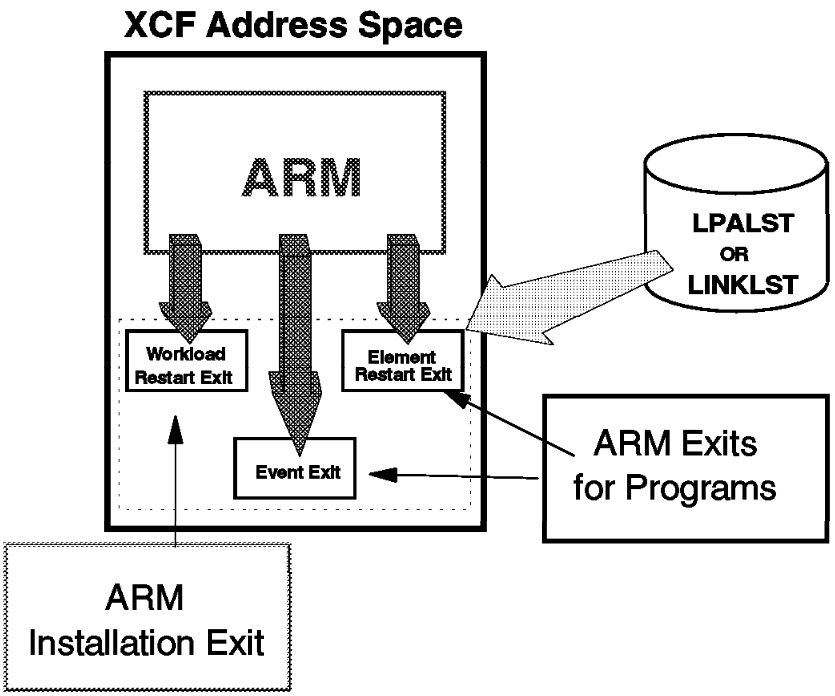


Figure 98. ARM exit facilities

5.1.11 ARM exit facilities

The exits shown on the visual are:

1. The workload restart exit (IXC_WORK_RESTART)

   This exit is executed once on each system where ARM is about to restart elements that had been running on a system that terminated. You might provide a routine for this exit to cancel low-priority work in preparation for the additional workload from the failed system that ARM is about to restart.

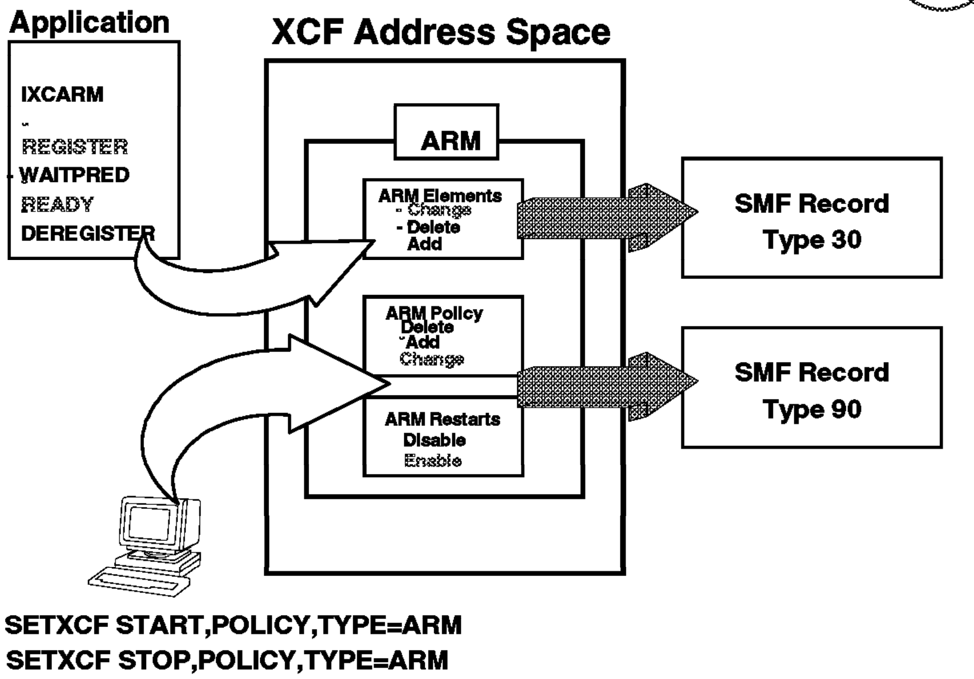
2. The element restart exit (IXC_ELEM_RESTART)

   This exit is executed before ARM restarts an element. It is used to coordinate restarts with the automation packages you may have installed.

3. The event exit

   This exit is usually provided by a program that is registering to become an ARM element. ARM invokes this exit for certain events that affect the element. For example, you could use the ENFREQ macro to listen for the deregister ENF signal for an element, then restart the element. ARM issues an asynchronous ENF signal, event-code 38, to communicate the occurrence of certain events.

ARM SMF Records



Application

IXCARM
-
REGISTER
WAITPRED
READY
DEREGISTER

XCF Address Space

ARM

ARM Elements
- Change
- Delete
Add

ARM Policy
Delete
Add
Change

ARM Restarts
Disable
Enable

SMF Record
Type 30

SMF Record
Type 90

SETXCF START,POLICY,TYPE=ARM
SETXCF STOP,POLICY,TYPE=ARM

Figure 99. ARM SMF records

5.1.12 ARM SMF records

The Type 30 (common address space work) and Type 90 (system status) SMF records have been enhanced to provide information relating to ARM activity, elements, and policy management.

The *SMF Type 30* record has been altered in these areas:

- Header section
- Product section
- ARM section

The new ARM section contains element identification information and time stamps for when various events complete. If the time stamp for an event is zero, it usually indicates that the event failed. This new section is created each time an element:

- Is started or restarted - The element issued an IXCARM REGISTER request.
- Is explicitly waiting for other elements to become functional - The element issued an IXCARM WAITPRED request.
- Is functional - The element issued an IXCARM READY request.
- Is no longer functional - The element issued an IXCARM DEREGISTER request during normal termination.

As each record is generated, the ARM time stamps that have been collected to that point will be included in the record. If a started task or job running in an address space has not requested an IXCARM REGISTER, then there will be no ARM section in the record. However, once the IXCARM REGISTER is requested, the ARM section will appear in every future Type 30 record that is generated for that started task or job.

The *SMF Type 90* record helps to assist in tracking the use of ARM policies. ARM generates a Type 90, subtype 27, 28, 29, or 30 record when one of the following events occur:

- An ARM policy is defined (new or changed) on the couple data set
- An ARM policy is deleted from the couple data set
- ARM is enabled for restarts via the SETXCF START (and a policy may be activated or changed)
- ARM is disabled for restarts via the SETXCF STOP (and a policy may be deactivated)

The contents of the SMF Type 90 records include:

- A function code identifying the event related to this record
- The name of the ARM policy that was affected:
  - The policy, if any, that was activated via a SETXCF START command
  - The policy, if any, that was deactivated via a SETXCF STOP command
  - The policy that was either defined (new or changed) or deleted from the ARM couple data set
- The name of the user who performed this function

ARM creates an SMF Record Type 90 (with new subtypes) to provide an audit trail when ARM restarts are enabled or disabled or a new policy is activated. This enables you to determine what ARM policy was active across a given interval if a question arises about an ARM action during that interval.

Chapter 6. Hardware management console

The hardware management console was introduced in the first 9672 Parallel Transaction Servers. It is now used for the complete line of 967x processors and can also be used with the 9221 211-based processors.

The main functions of the HMC are:

- Single point of operations for multiple systems.
- TCP/IP SNMP agent and application programming interface (API).
- Drag and drop for functions such as IPL, reset, and load.
- DCAF (single object operations) to the 9221 console eliminating the need for working near the machine. The 3270 sessions provided with FC2000 can also be used in DCAF mode.
- Better interface for console integration.
- Message routing.

HMC

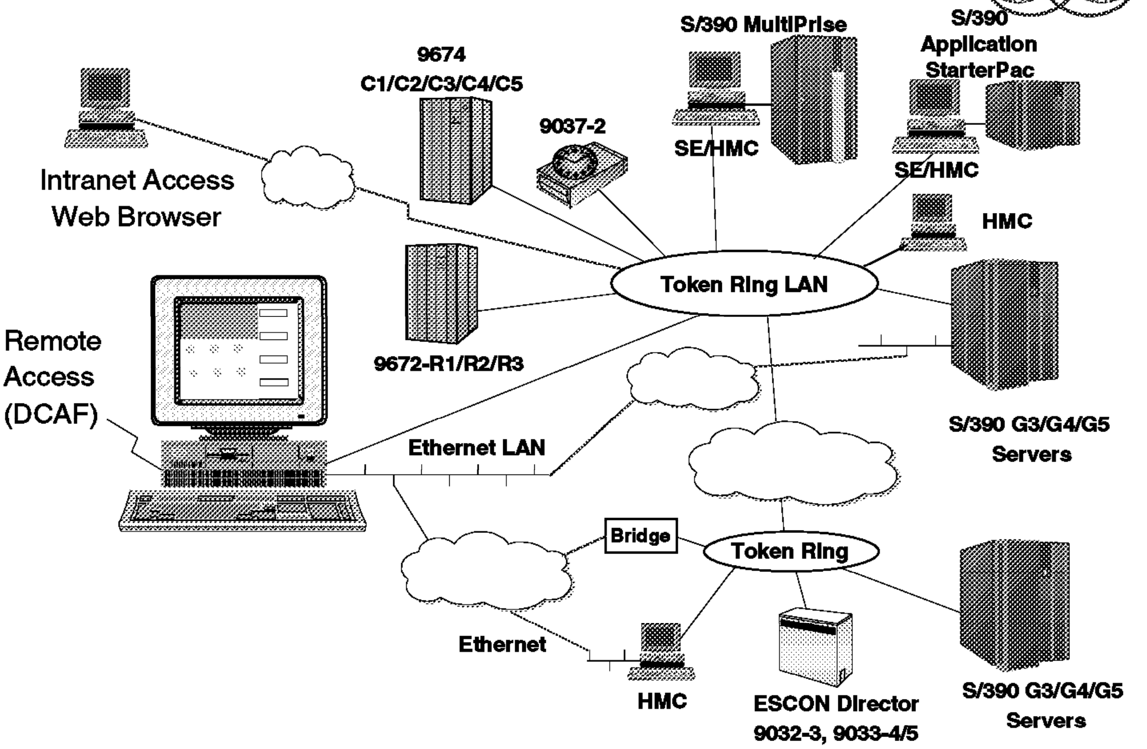


Figure 100. Hardware management console (HMC)

6.1 Hardware management console (HMC)

The hardware management console provides a single point of control to manage your central processor complex (CPC). It uses a GUI interface and is connected to your CPC via a local area network (token ring or ethernet). HMC provides:

- Full operation of managed processors and images
- Real-time system status reporting via icons and colors
- Consolidates:
  - Hardware messages
  - Operating system messages
  - Service support (including RSF)

The hardware management console communicates with each central processor complex (CPC) through the CPC′s support element (SE). When tasks are performed at the hardware management console, the commands are sent to one or more support elements which then issue commands to their CPCs. CPCs can be grouped at the hardware management console so that a single command can be passed along to as many as all of the CPCs defined to the hardware management console.

What is an HMC

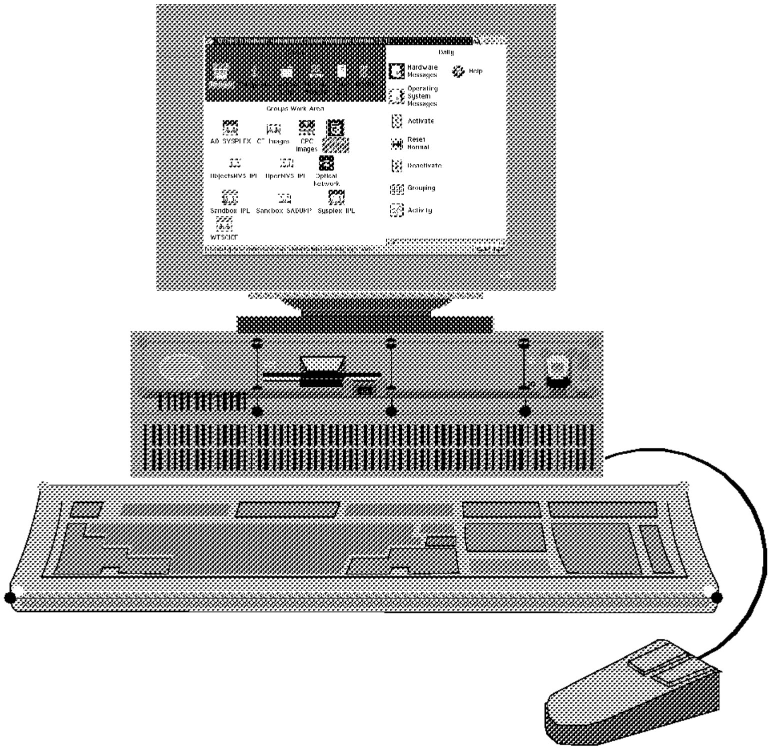


Figure 101. What is an HMC

6.2 What is an HMC

The HMC is a specially configured PC (both hardware and software) running with OS/2 as the operating system and the HMC application installed. The HMC application is started automatically at power on. An HMC can activate, operate, monitor, and perform recovery tasks on up to 32 CECs or CFs. Up to 15 HMCs may be on the same local area network. The HMC can also be used to control ESCON directors 9032 models 3, 4, and 5 and the model 2 Sysplex Timer.

6.2.1 Online help

The HMC application includes comprehensive help panels. They provides both general and specific information. Any icon can be dragged and dropped on the Help icon for information, or the Help icon can be dragged and dropped on any of the icons in the views, tasks, or work areas of the Hardware Management Console Workplace window.

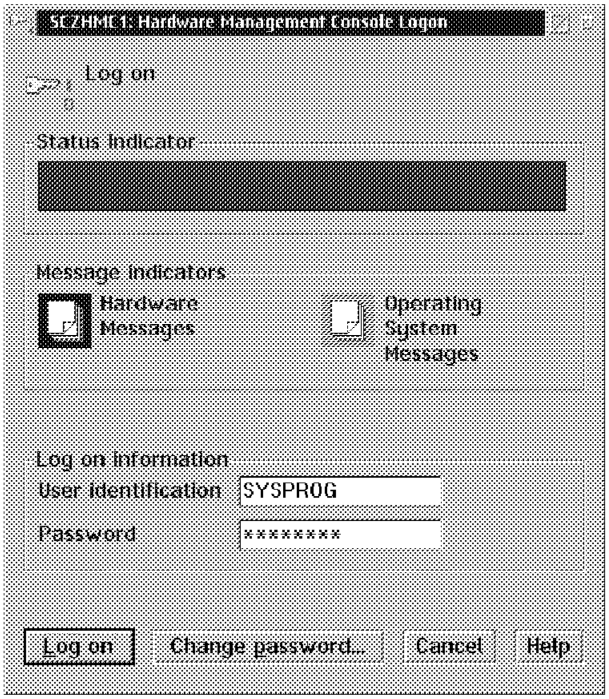
To display help for an object or hardware management console area drag and drop the Help icon on the object or the area of the hardware management console that you want help information for.

The Help window displays help information for the object or area of the hardware management console where you dropped the Help icon.

Help is also available for each window that is displayed by selecting the F1 key on the keyboard. Depending on the type of window, the help displayed will either be field help or the general help. If the window has one or more fields, the help displayed will be for the field where the cursor is located. To display the general help for the window, select F2 while the field help is displayed. If the window does not have fields, the general help for the window is displayed when F1 was selected.

Most windows will also provide a Help push button which may be selected with the mouse.

Logon Screen



User identification SYSPROG

Password ********

Figure 102. Logon

6.2.2 HMC logon

Before you can use the HMC console you must log on. The logon window is displayed after initialization is complete or after the current user is logged off.

The default user IDs and passwords are:

Table 1. Default user IDs and passwords

Authority mode	Default USERID	Default password
Access Administrator	ACSADMIN	PASSWORD
System Programmer	SYSPROG	PASSWORD
Advanced Operator	ADVANCED	PASSWORD
Operator	OPERATOR	PASSWORD
Service	SERVICE (used by service representative)	SERVMODE

6.2.2.1 Access Administrator

An access administrator can use the HMC to:

- Create user IDs
- Assign passwords
- Name CPCs
- Name HMCs
- Define CPCs to HMC
- Customize task lists

6.2.2.2 System Programmer

A system programmer can use the HMC to:

- Build, delete, and customize profiles
- Activate CPC and image
- Operate CPC and image
- Monitor CPC image
- Perform recovery tasks

6.2.2.3 Advanced Operator

An advanced operator can use the HMC to:

- Activate CPC and image
- Operate CPC and image
- Monitor CPC image
- Perform recovery tasks

6.2.2.4 Operator

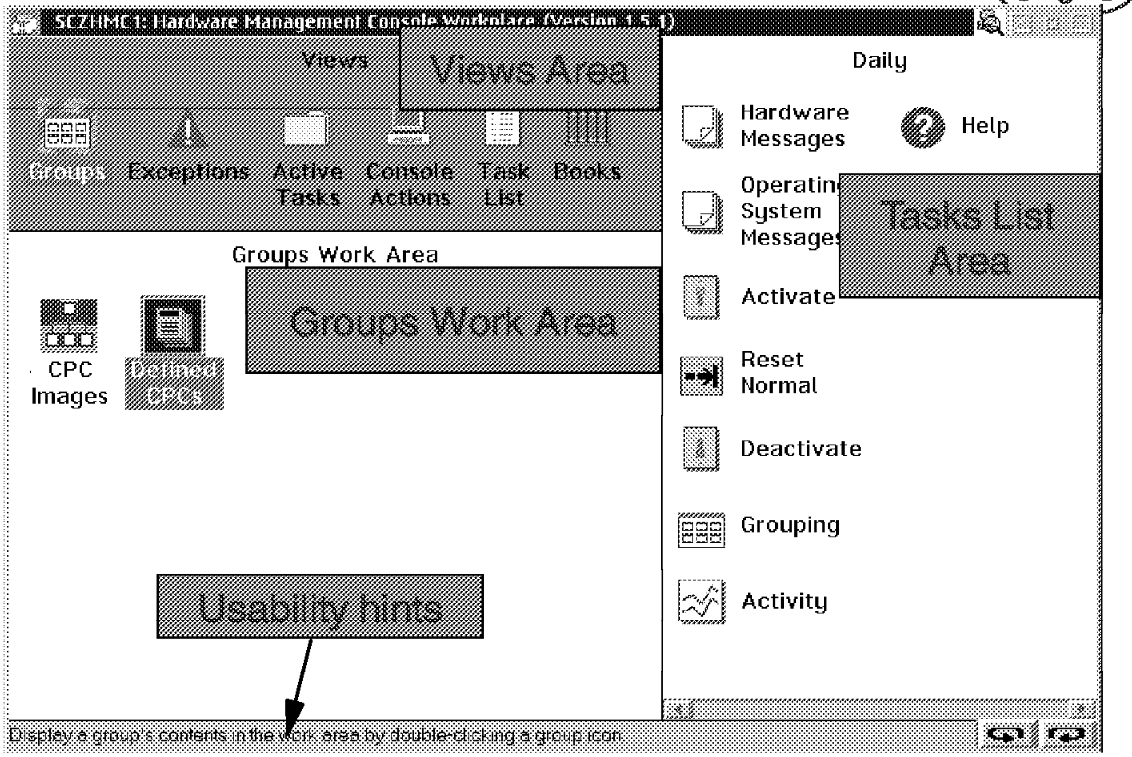
An operator can use the HMC to:

- Activate CPC and image
- Operate CPC and image
- Monitor CPC image

6.2.2.5 Service

Service personnel can use the HMC to perform service and maintenance tasks.

Logon Complete



Display a group's contents in the work area by double-clicking a group icon

Figure 103. Logon complete

6.3 HMC panel displays

After logon is complete the HMC Workplace panel is displayed.

The panel has three areas:

- Views area
- Work area
- Task list area

6.3.1 Views

Views of the processor complex objects are represented in the Views area. Each view provides a different way of looking at the processor complex objects. After an object is opened (double-clicking on the object in the Work Area opens the object to another level of detail, if available), it is displayed in the Work Area and is available for further action.

6.3.2 Work Area

The Work Area displays the objects of your processor complex based on the object in the Views area that you open (double-click). Objects must be displayed in the Work Area before tasks can be performed on them. The icons represented in Views are:

6.3.2.1 Groups

Groups are displayed initially when you log on and Groups Work Area is the view that will probably be used most often to run and monitor the processor complex. Groups contain logical collections of objects. They provide a quick and easy means for performing tasks against the same set of objects more than once, eliminating the need of having to select each object every time the task is run. Status is reported on a group basis allowing an installation to monitor the system. One of the group icons can be selected and a task can be performed on the objects in the group by dragging a task icon to one of the group icons or dragging one of the groups to a task icon [place the mouse pointer on the icon, press and hold down the right mouse button, then move the mouse (and icon (drag an object) to another location, release the object (drop the object) by releasing the right button].

The system-defined groups are provided with the system and consist of all CECs or all images that make up the processor complex. User-defined groups can be created from these groups (using daily tasks) to represent views of CECs or images that the installation wishes to work with:

- *Defined CPCs* are all of the CECs configured to the hardware management console
- *CPC Images* are all the images in the processor complex (in LPAR an image is a partition where an operating system is running; in non-LPAR it is the system itself)
- *Undefined CPCs* are all CECs that are physically installed but not configured to the hardware management console

6.3.2.2 Exceptions

If an exception exists, either the Views portion of the console will have a red background or the background of the exception object will be red. An exception state occurs as a result of an unacceptable status condition. Double-click on the exceptions icon using the left mouse button and the exception objects representing all CECs and images that are in an exception state will be displayed in the Work Area. To take corrective action, select one of the icons in the exception work area and perform tasks on the object.

6.3.2.3 Active Tasks

Active Tasks allows an installation to view the object icons for all the tasks that are in progress or those tasks that have completed but whose ending status has not been reviewed.

6.3.2.4 Console Actions

Console Actions is the set of operations that can be performed on the hardware management console, including setting up the hardware system console, maintaining its internal code, and servicing it:

- View console events
- View console service history
- Customize date/time
- Change internal code
- Back up critical console data
- Authorize remote connections
- Customize automatic logon
- User profiles
- Customized product engineering access
- Hardware management console settings

6.3.2.5 Task List

The Task List area of the hardware management console is initially displayed with the Daily Tasks icons in the task area on the right portion of the panel. To change to a different set of tasks, double-click on one of the icons in the tasks view. Each task list contains three common icons; hardware messages, operating system message, and help. The types of tasks are:

- Daily tasks
- Recovery tasks
- Service tasks
- Remote customization tasks
- Configuration tasks
- Change Management tasks
- CEC Operational customization tasks
- Object definition tasks

6.3.2.6 Books

The Books icon displays an online version of the hardware system console publication.

6.3.3 Mouse and keyboard usage

To select an item on a panel, move the mouse pointer to the item and *click* the left button of the mouse. Double-click refers to the practice of clicking the button twice in succession. Dragging an icon is accomplished by placing the mouse pointer on the icon and pressing and holding down the right mouse button while dragging the icon to another location. Dropping an icon is accomplished by releasing the right mouse button when the icon is at its new location.

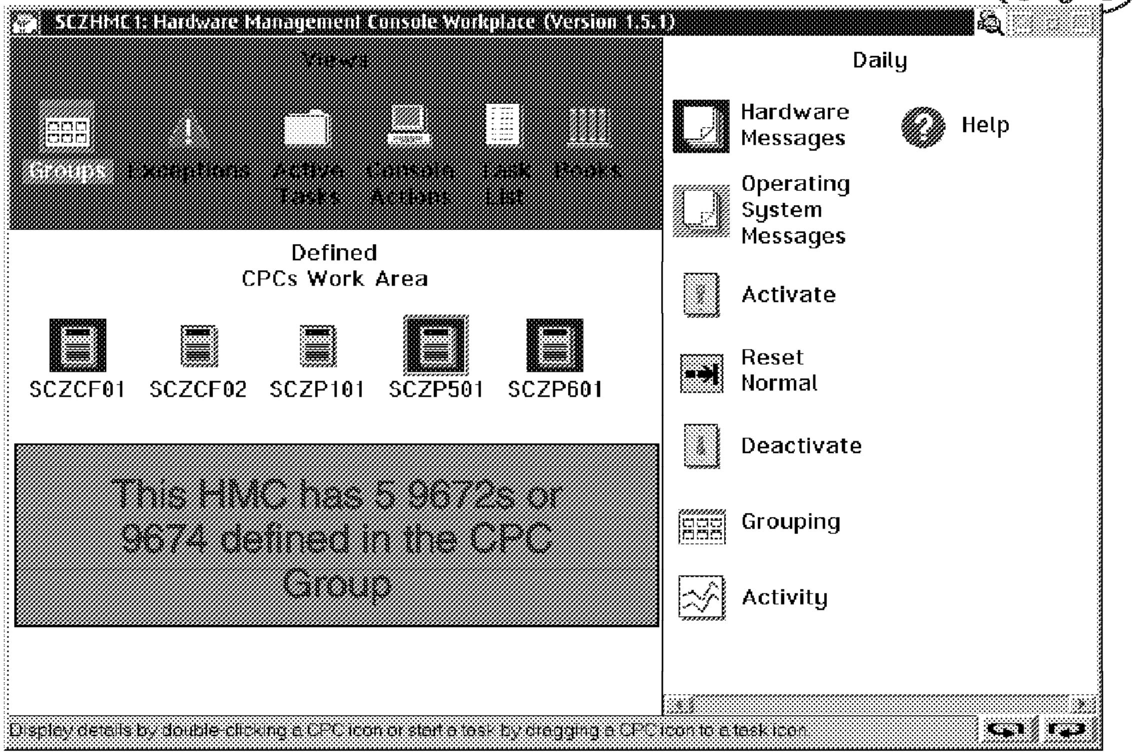
Color is used to indicate the status of the CECs on the Hardware System Console Workplace panel. The default colors are:

Green Green indicates good, or operational status of all CECs and CEC images.

Red Red indicates exception, or not-operational status of one or more CECs or CEC images.

Blue Waiting messages are indicated by a flashing blue HMC icon or a flashing cyan Operating System Messages icon in the task area.

Defined CPCs Group Open



This HMC has 5 9672s or 9674 defined in the CPC Group

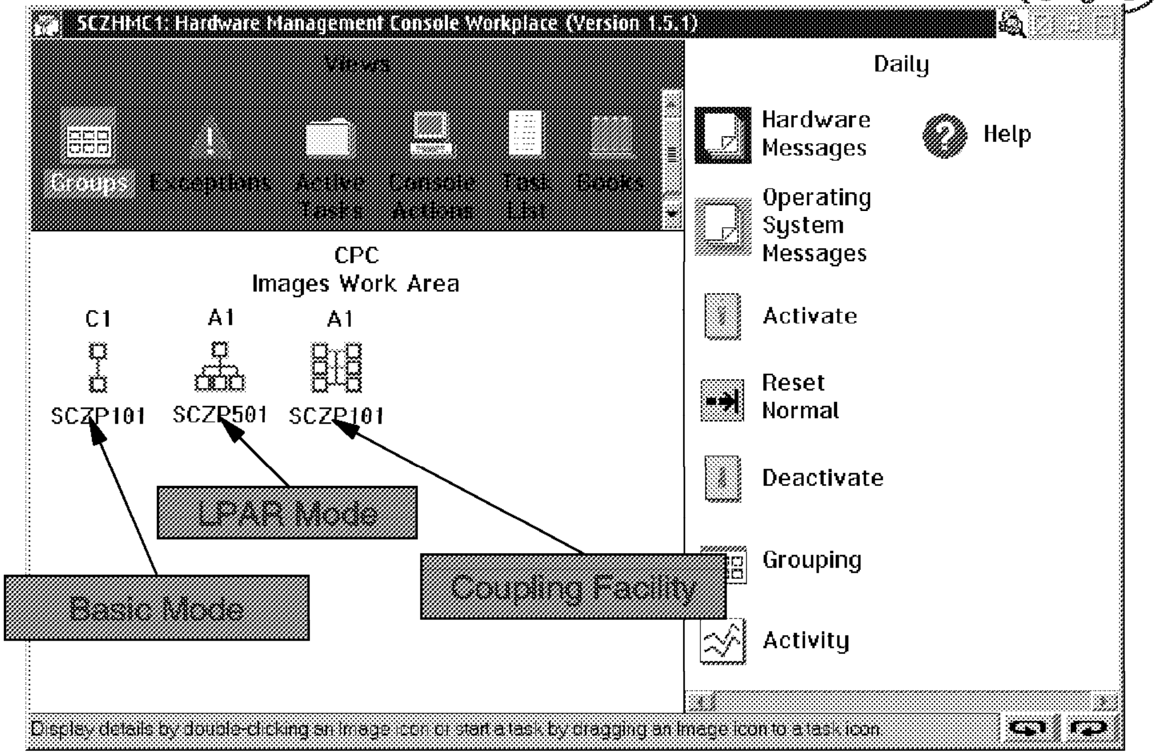
Display details by double clicking a CPC icon or start a task by dragging a CPC icon to a task icon

Figure 104. CPC group

6.4 CPC group

Double clicking on the Defined CPCs group will display the CPC objects defined to this HMC. In this example there are five CPCs defined to this HMC. Double clicking on any one of the CPC objects will show more detail on this object.

CPC Images Group Opened



Display details by double-clicking an image icon or start a task by dragging an image icon to a task icon.

Figure 105. Image group

6.5 Image group

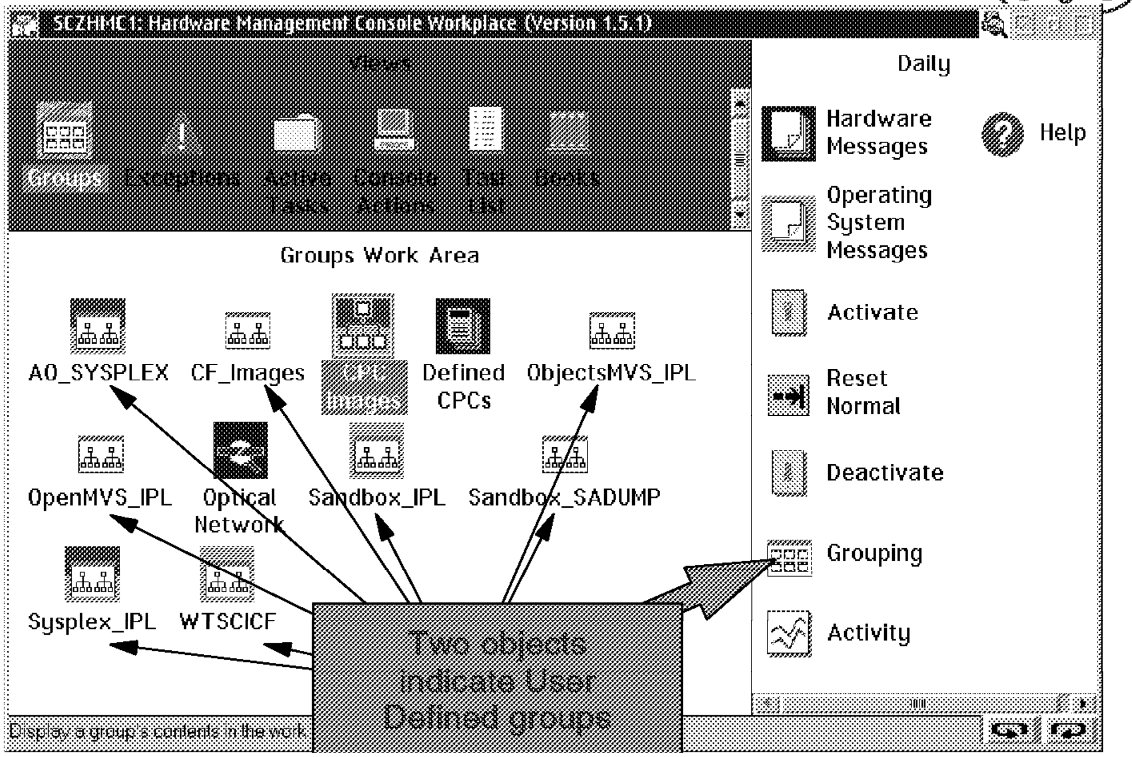
The operating mode of any image can be determined by the *object* assigned to that Image. There are three different objects for:

- Basic mode
- LPAR mode
- Coupling Facility

Objects are only present in the Images group if POR is complete and one of the following conditions is not met:

- Service status enabled
- Communications not active
- Status check (SE to CPC communication)

User Built Groups



Two objects indicate User Defined groups

Figure 106. User-built groups

6.6 User-built groups

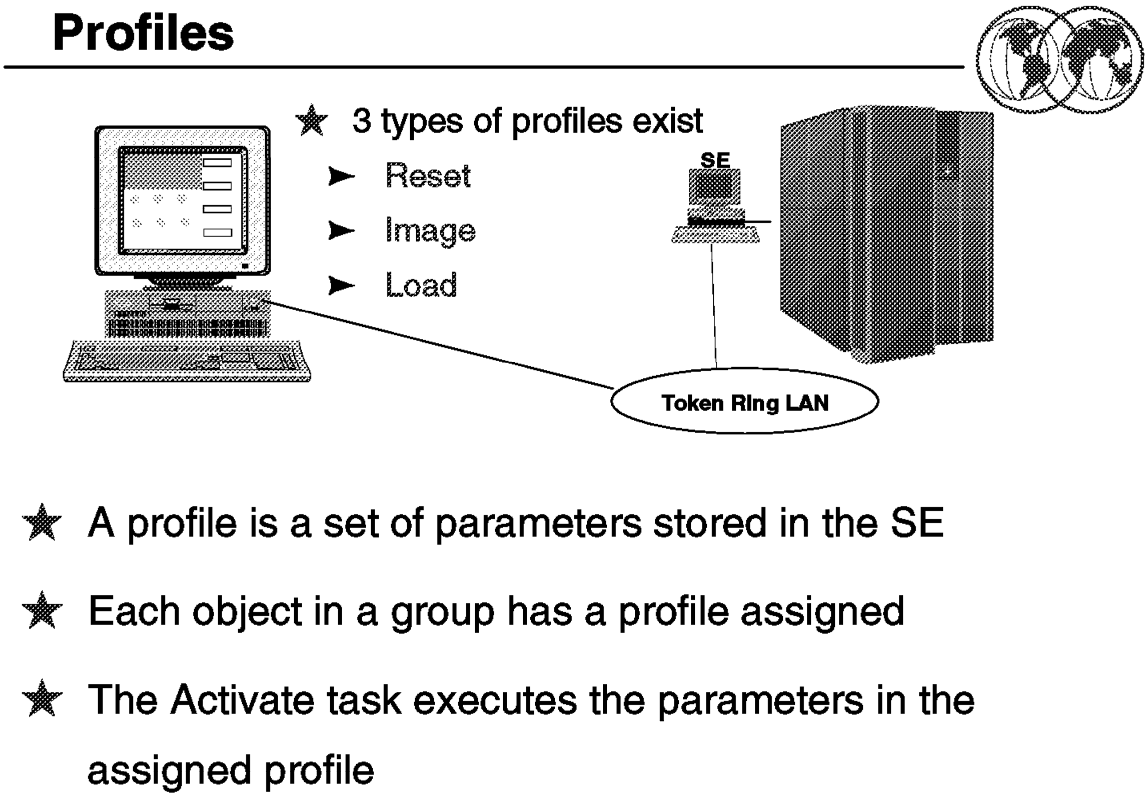
A user-built group is a logical collection of objects. You may want to create a group to perform the same task on several CPCs or CPC images simultaneously instead of repeating the task on each individual CPC or CPC image.

You may also want to create groups when managing multiple sysplexes by creating a group for each sysplex controlled by the hardware management console. As you can see it is a way of grouping like functions together. Each group should have a meaningful name.

Two types of groups can be built, modified, or deleted:

- CPC
- Images
  - Objects in the CPC groups control the hardware (CPC).
  - Objects in the Image groups control the operating systems.
  - Activation profile is assigned to each object in the group:
    - Used to power up
    - Power-on reset (POR)
    - Activate LPAR
    - IPL

- Require SYSPROG or ACSADMIN ID to build a new group



Profiles

- 3 types of profiles exist
  - Reset
  - Image
  - Load

- A profile is a set of parameters stored in the SE
- Each object in a group has a profile assigned
- The Activate task executes the parameters in the assigned profile

Figure 107. Profiles

6.7 Profiles

To make your 9672 Parallel Transaction Server, 9672 Parallel Enterprise Server, and 9674 Coupling Facility easier to use, frequently-used sequences of functions have been combined into the task of *Activate*. The Activate task brings up your 9672 Parallel Transaction Server, 9672 Parallel Enterprise Server, and 9674 Coupling Facility, which means the activation:

- Powers on the system hardware if it was powered off
- Completes a power-on reset (called an IML on previous systems)
- Activates the logical partitions (if in LPAR mode)
- Loads the operating system (called an IPL on previous systems)

Instead of performing three separate functions, the single task, Activate, performs these functions for you.

To accomplish the Activate task, parameters must be defined for the Reset and Load profiles for ESA/390 modes. If you are using logically partitioned mode, parameters must be defined for the Reset and Images profiles.

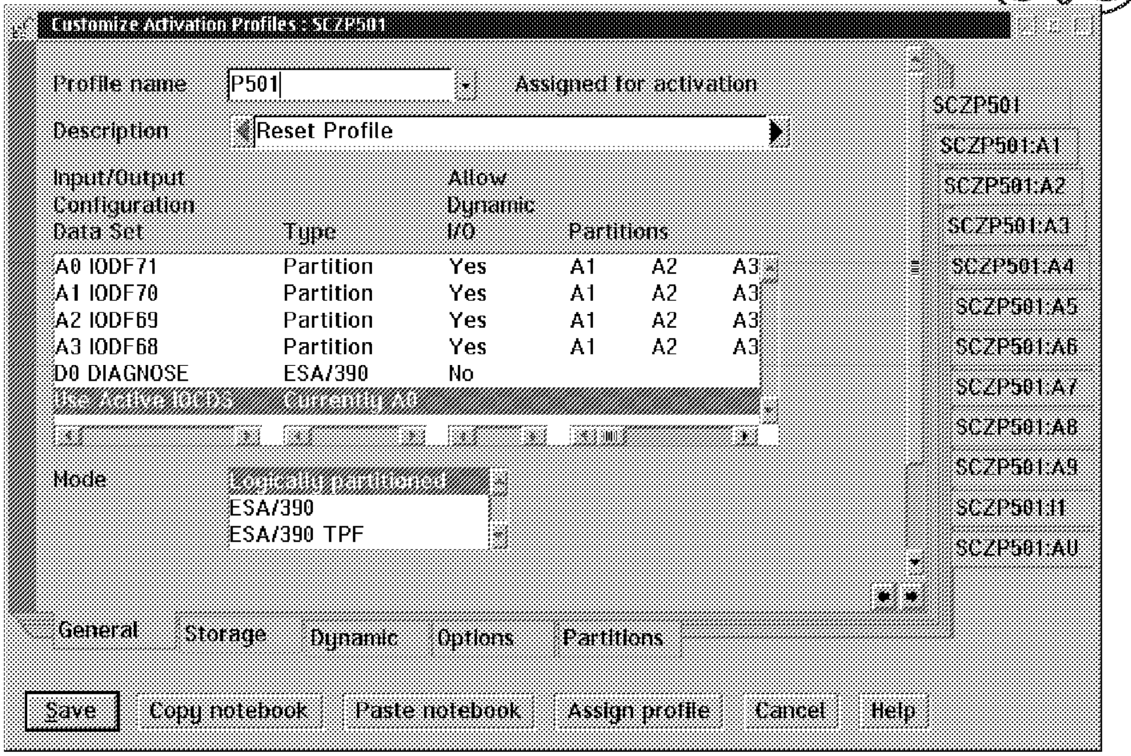
The Load profile defines parameters for the processing unit, load device address, and associated parameters.

The Reset profile defines the IOCDS, the parameters to power-up and power-on reset, and also identifies the Load or Image profiles that will be used.

The Image profile defines parameters to control the activation of logical partitions.

A set of default activation profiles is shipped with the system. The values contained in these profiles may not be correct for your environment. Although the system lets you change the default activation profiles, it is recommended that you copy them into a new profile, modify the new profile with customized parameter values, and rename the profile for your own applications. You should define a set of profiles for each way that you plan to start up your system; for example, set up separate profiles for a normal start up, an alternate load device, and the load of a dump program.

Reset Profile



Customize Activation Profiles : SCZP501

Profile name: P501 — Assigned for activation

Description: Reset Profile

Input/Output Configuration Data Set	Type	Allow Dynamic I/O	Partitions		
A0 IODF71	Partition	Yes	A1	A2	A3
A1 IODF70	Partition	Yes	A1	A2	A3
A2 IODF69	Partition	Yes	A1	A2	A3
A3 IODF68	Partition	Yes	A1	A2	A3
D0 DIAGNOSE	ESA/390	No			
Use Active IOCDS	Currently A0				

Mode: Logically partitioned, ESA/390, ESA/390 TPF

Figure 108. Reset profile

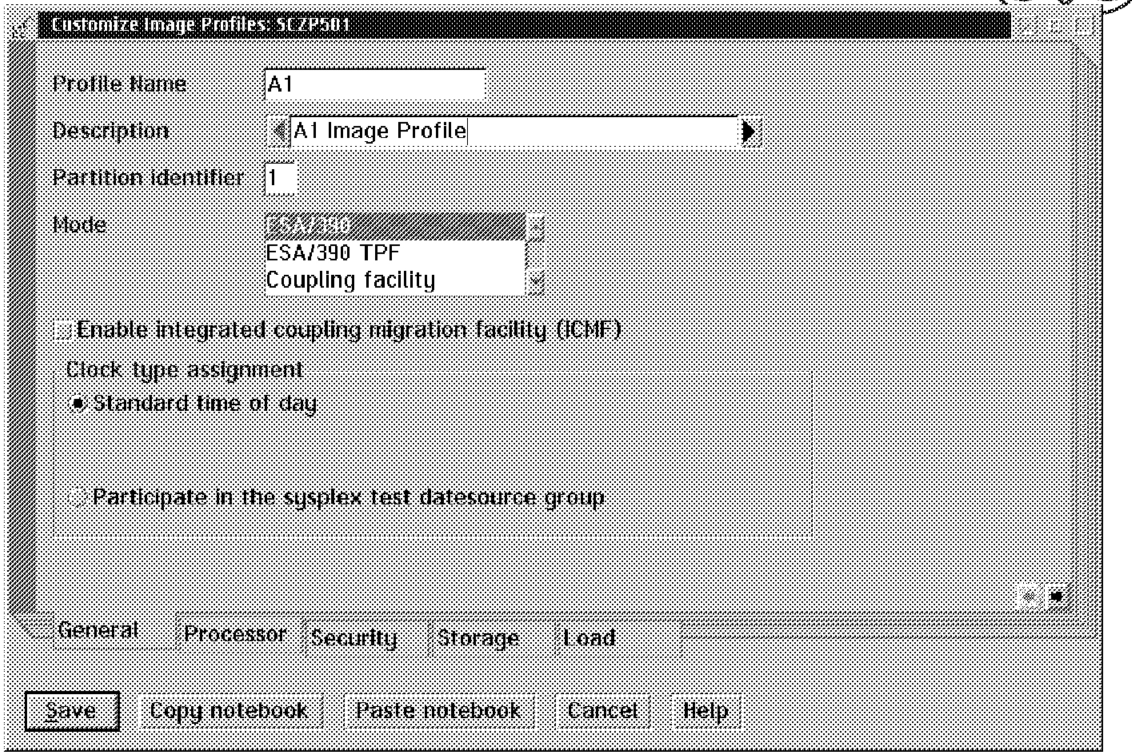
6.8 Reset profile

Every CPC in the processor cluster needs a reset profile to determine the mode in which the CPC licensed internal code will be loaded.

Function

- Power up the CPC or CF if not already powered up
- Power-on reset (POR)
- Defines IOCDS
- Mode of operation
- Storage assignments
- Dynamic I/O
- I/O reset
- Partition activation order if operating in LPAR mode
- Point to a load profile if operating in Basic mode

Image Profile



Customize Image Profiles: SCZP501

Profile Name: A1

Description: A1 Image Profile

Partition identifier: 1

Mode: ESA/390 (selected), ESA/390 TPF, Coupling facility

□ Enable integrated coupling migration facility (ICMF)

Clock type assignment

● Standard time of day

○ Participate in the sysplex test datesource group

Figure 109. Image profile

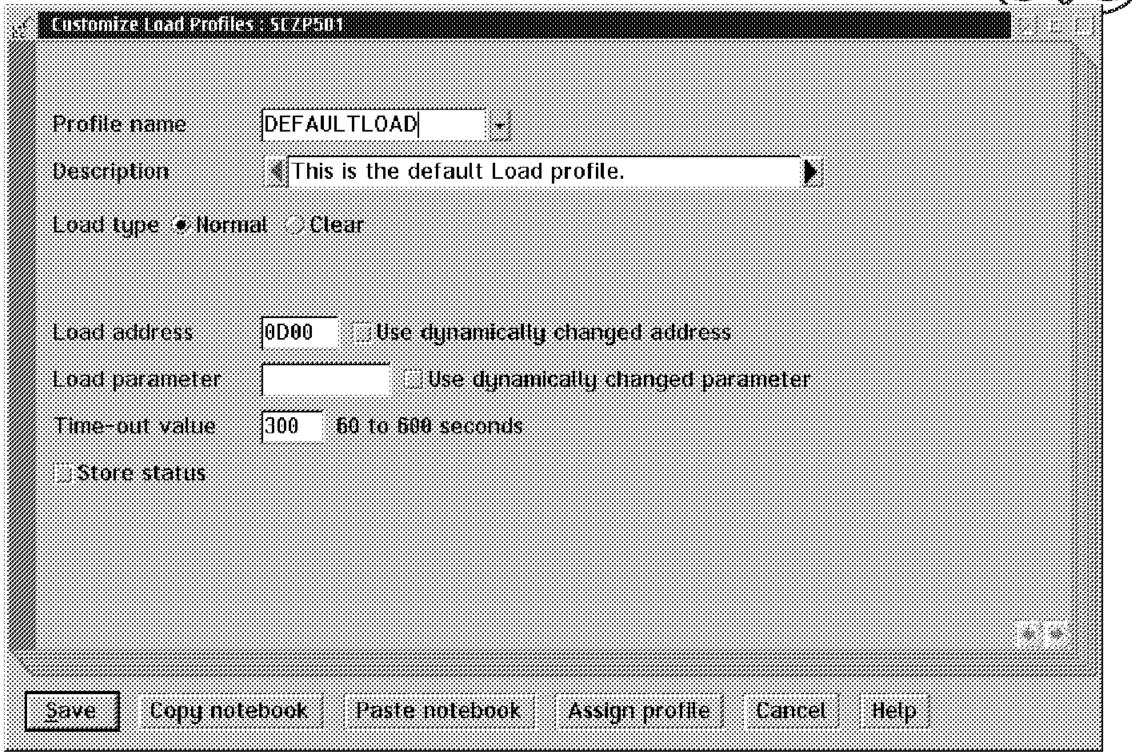
6.9 Image profile

Defines the parameters required to activate each LPAR (partition).

Function

- Defines mode of operation
- Partition ID
- Numbers of CPs shared or dedicated
- Storage assignments central and expanded
- Security options
- Processor weight for non-dedicated processors
- Cryptographic coprocessor
- Sysplex Timer offset
- Asynchronous data mover (ADM)

Load Profile



Profile name DEFAULTLOAD

Description This is the default Load profile.

Load type ● Normal ○ Clear

Load address 0D00 □ Use dynamically changed address

Load parameter □ Use dynamically changed parameter

Time-out value 300 60 to 600 seconds

□ Store status

Figure 110. Load profile

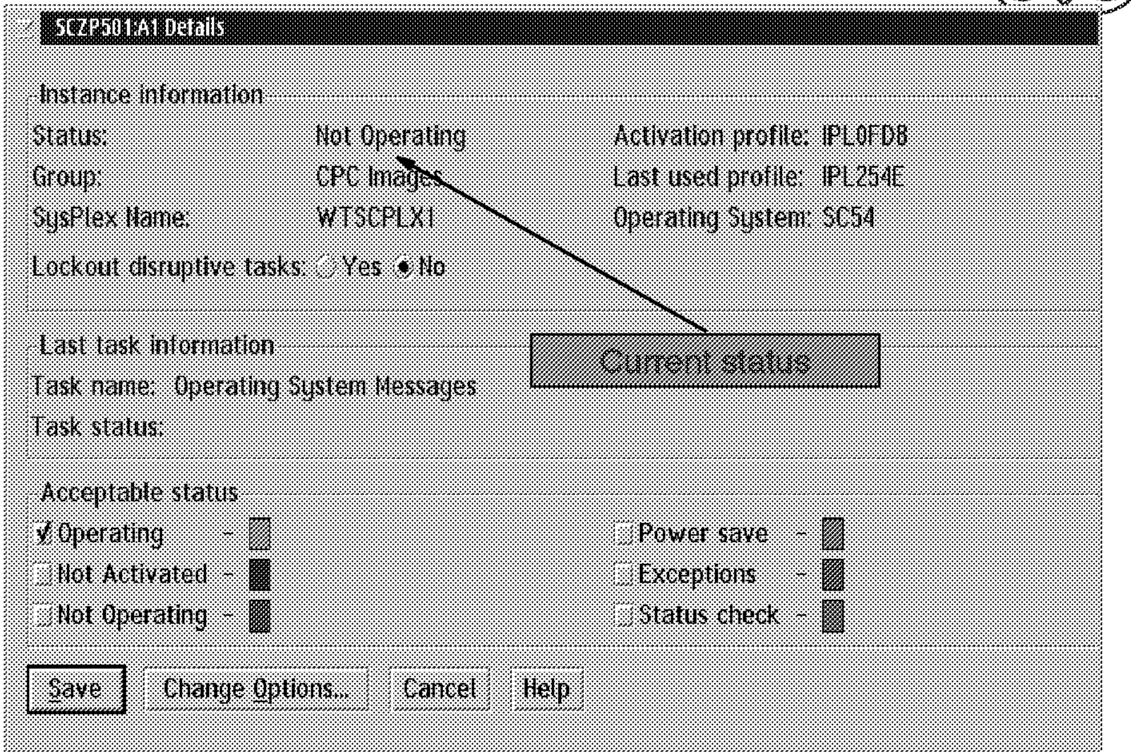
6.10 Load profile

A load profile is needed to define the channel address of the device that the operating system will be loaded from. The maximum number of load profiles allowed for each CPC is 64. If the dynamically changed address and or parameter box is checked the load address and or parameters are obtained from HCD.

Function

- Defines the type of load (normal or clear)
- IPL load device number
- Load parameters

Image Details



SCZP501:A1 Details

Instance information

Status: Not Operating

Group: CPC Images

SysPlex Name: WTSCPLX1

Lockout disruptive tasks: ○ Yes ● No

Activation profile: IPL0FD8

Last used profile: IPL254E

Operating System: SC54

Last task information

Task name: Operating System Messages

Task status:

Acceptable status

☑ Operating – ▨

□ Not Activated – ▨

□ Not Operating – ▨

□ Power save – ▨

□ Exceptions – ▨

□ Status check – ▨

Current status

Figure 111. Image details

6.11 Status conditions

The Details panel provides the current status of the object in the upper left hand corner. This field is dynamically updated (instant information).

Opening either an Image or CPC icon will display the Details panel.

Acceptable status conditions are indicated in the checked boxes (these conditions will not cause an exception).

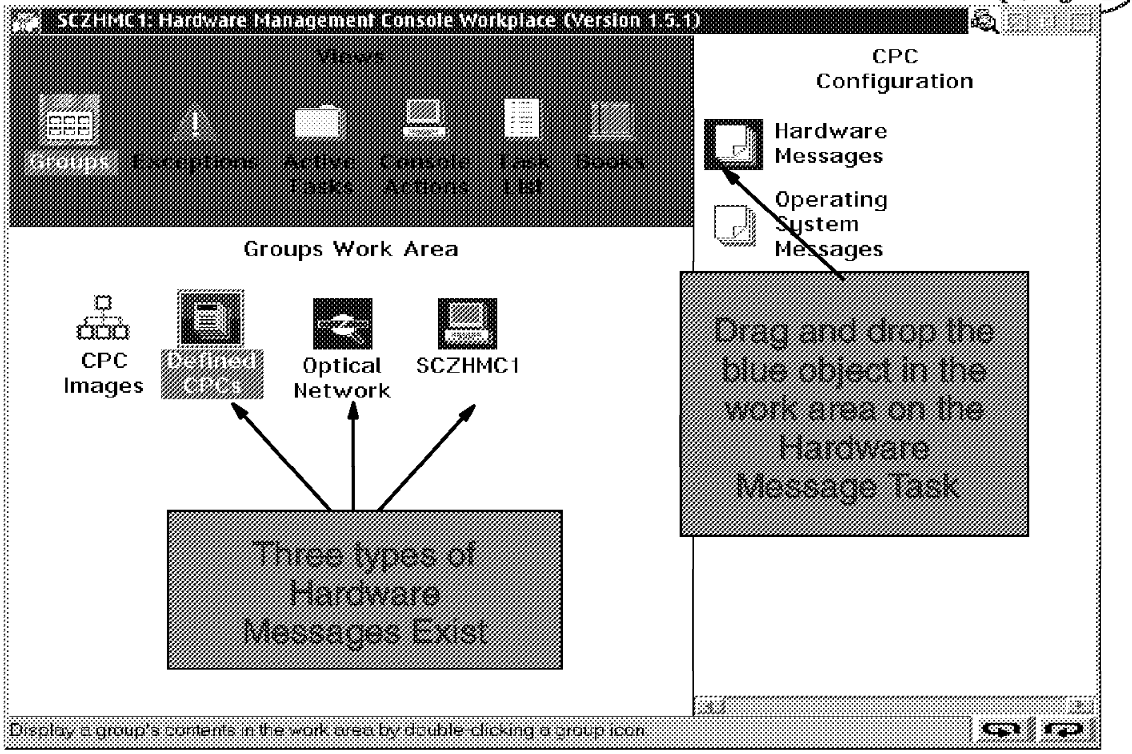
Non-acceptable status is indicated by the lack of a check in the box next to the condition (these conditions will cause an exception).

The color for each status condition is to the right of the text.

Table 2. Status conditions

Status condition	Description
Operating	CPC image is operational
Not operating	CPC or image has been system reset CPC or Image is in the disabled wait state CPC is powered up but not POR complete
No power	CPC power is off
Power Save	Power save is active—external power lost
Exceptions	PUs (CPs) are in a mixed status—one or more PU is operating while one more PUs is in manual, offline or wait state
Status Check	Communication problem from the SE to the CPC
Communications Not Active	HMC cannot communicate with the SE
Service	CPC is in service mode
Not Activated	LPAR partition is deactivated

Hardware Messages



Drag and drop the blue object in the work area on the Hardware Message Task

Three types of Hardware Messages Exist

Display a group's contents in the work area by double-clicking a group icon.

Figure 112. Hardware messages

6.12 Hardware messages

Hardware Messages displays consolidated hardware-related messages for all selected hardware in the S/390 microprocessor cluster, including your hardware management console.

When a message is logged, the Hardware Messages icon alternates between a light and dark background. The hardware object (CPC or hardware management console) with a message pending is indicated by a dark background on its object icon. By default, the dark background color of the object icon and the Hardware Messages icon is blue.

A message is a brief, one-line description of an event, such as a report of a hardware management console failure. Further explanation and any recommended operator action for a single message may be viewed by double-clicking on the message. Or you may select one or more messages, then select Details. The message details and any recommended operator action display, one at a time, for each selected message.

Hardware messages for all of the hardware objects are stored on the hardware management console hard disk in the message log file. Because the message log file limits the number of messages to five hundred, try to view, act on, and delete messages promptly.

Messages received over this limit will cause the oldest messages to be lost. Delete selected messages from the list by selecting Delete. A panel is displayed where older messages are viewed that can be acted on or deleted.

Note: Some messages are deleted automatically after you view the message details. These messages generally provide information only, and are deleted automatically because no further action is required.

There are three types of hardware messages and a new hardware message is indicated by the Hardware Messages icon (Task List) flashing blue (default color).

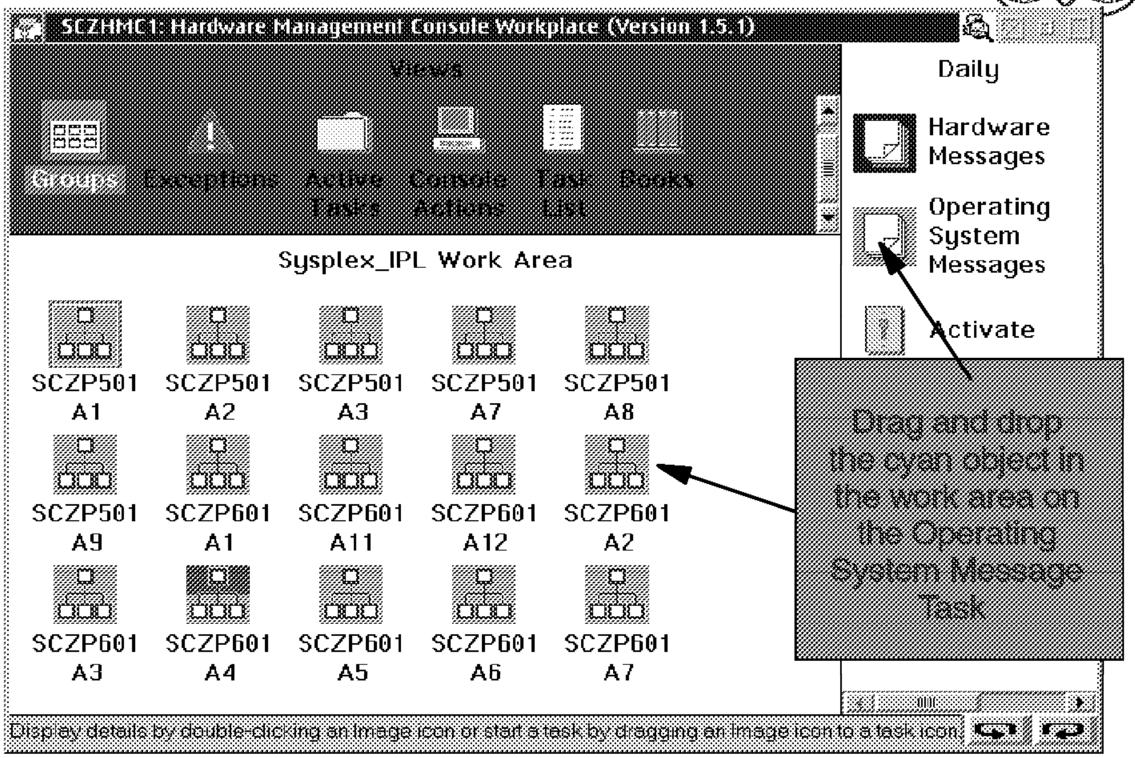
The flashing blue Hardware Messages icon could be caused by:

- A problem reported against a CPC or CF in a Defined CPC group or user-built CPC group.
- A problem reported by the hardware management console.
- A problem with the optical network (ESCON channels, Coupling facility links).

To view the hardware message, drag and drop the blue object in the Work Area on the Hardware Messages Task.

Note: The icons for the hardware management console and the optical network are only present in the Groups Work Area when messages are present.

Operating System Messages



Drag and drop the cyan object in the work area on the Operating System Message Task

Display details by double-clicking an image icon or start a task by dragging an image icon to a task icon.

Figure 113. Operating System Messages

6.13 Operating System Messages

The Operating system Messages task provides the ability to send a command to any active MVS operating system in the sysplex or to respond to a WTOR using the console integration function provided by the HMC and S/390.

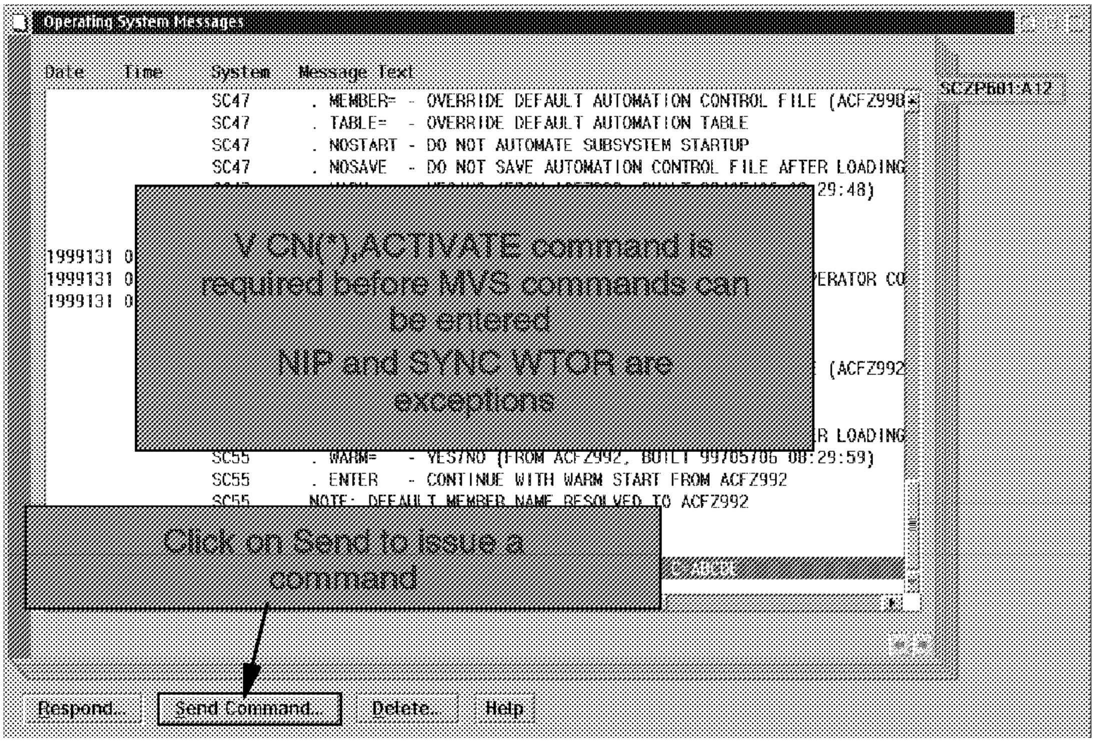
Since it is expected that some members of the S/390 Parallel Sysplex will run without channel-attached consoles, the Operating System Message task can be used to respond to an MVS message or send an MVS command.

The Operating System Messages icon is present on all the Task Area lists. Operating System Message displays consolidated operating system-generated messages for all selected CPC images. The operating system messages are displayed when one or more CPC image objects, or a group of CPC images, is dragged and dropped on the Operating System Messages icon.

Coupling Facility Control Code (CCC) commands can be sent from the hardware management console to a CF when a CPC image that represents a CF is dragged and dropped on the Operating System Messages icon. To send a command, click on the Send Command push button.

For more information about the CFCC commands and messages, refer to the online books, *Coupling Facility Control Code Commands* and *Coupling Facility Control Code Messages*.

Operating System Message Task Invoked



Operating System Messages

```
Date     Time      System   Message Text
                   SC47      . MEMBER= - OVERRIDE DEFAULT AUTOMATION CONTROL FILE (ACFZ998
                   SC47      . TABLE=  - OVERRIDE DEFAULT AUTOMATION TABLE
                   SC47      . NOSTART - DO NOT AUTOMATE SUBSYSTEM STARTUP
                   SC47      . NOSAVE  - DO NOT SAVE AUTOMATION CONTROL FILE AFTER LOADING
                                                                         29:48)
1999131 0
1999131 0                                                              ERATOR CO
1999131 0
                                                                       (ACFZ992

                                                                       R LOADING
                   SC55      . WARM=   - YES/NO (FROM ACFZ992, BUILT 99705706 08:29:59)
                   SC55      . ENTER   - CONTINUE WITH WARM START FROM ACFZ992
                   SC55      NOTE: DEFAULT MEMBER NAME RESOLVED TO ACFZ992
```

V CN(*),ACTIVATE command is required before MVS commands can be entered
NIP and SYNC WTOR are exceptions

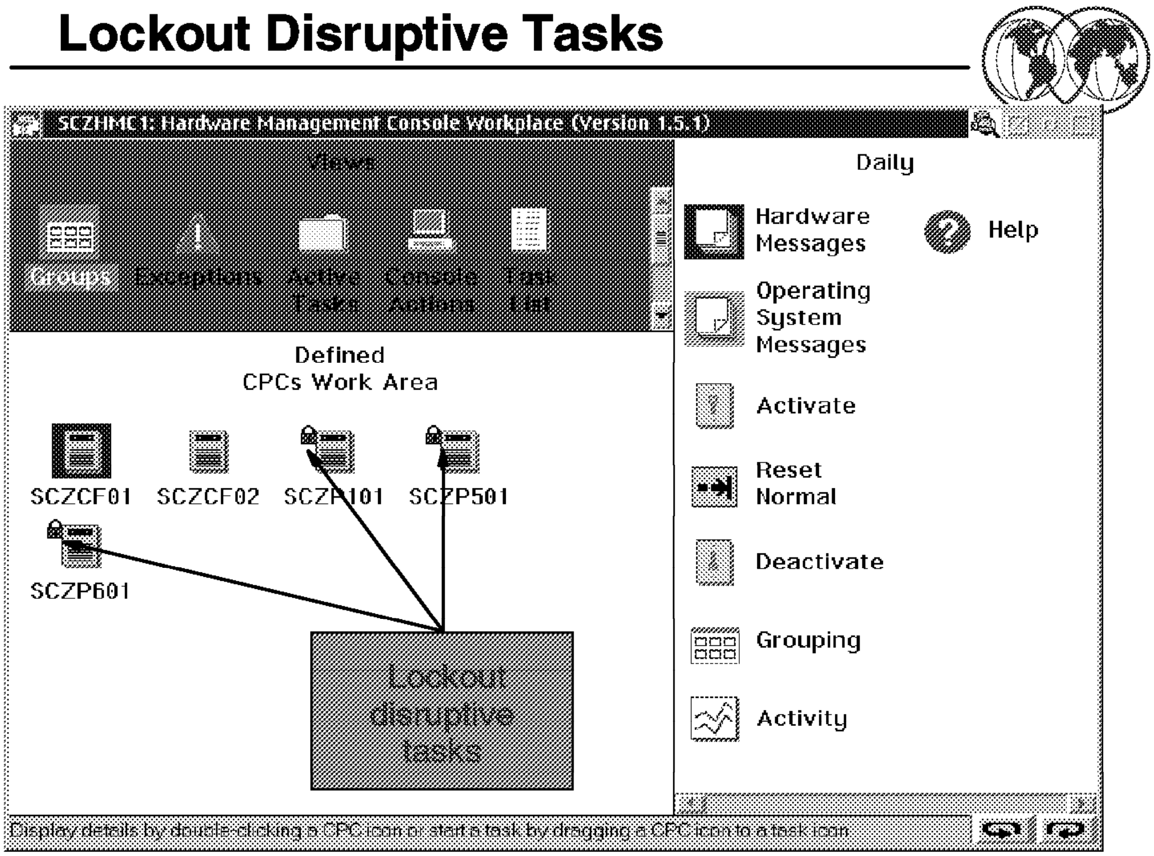
Click on Send to issue a command

Figure 114. Operating System Messages task invoked

6.14 Operating System Messages task invoked

The Operating System Messages task allows the sending of commands to MVS or a Coupling Facility.

After the master scheduler has been started during NIP the `V cn(*),activate` command needs to be entered prior to issuing MVS commands.



Lockout Disruptive Tasks

SCZHMC1: Hardware Management Console Workplace (Version 1.5.1)

Display details by double-clicking a CPC icon or start a task by dragging a CPC icon to a task icon.

Figure 115. Lock out disruptive tasks

6.15 Lock out disruptive tasks

Some of the hardware management console tasks are considered disruptive. Performing a disruptive task on a CPC or CPC image may disrupt its operation. You may want to lock an object to prevent accidentally performing a disruptive task on it and then unlock the object only when you want to perform a disruptive task on it. The following are considered disruptive tasks:

Daily tasks	Activate, Reset Normal, and Deactivate
CPC recovery tasks	Start, Stop, Reset Normal, PSW Restart, Reset Clear, and Load
Service tasks	Checkout Tests
Change management tasks	Engineering Changes (ECs), Change Internal Code, Product Engineering Directed Changes, and CPC Operational
Customization	Change LPAR Controls, Configure Channel Path On/Off, and Reassign Channel Path
Object definition tasks	Change Object Definition and Reboot Support Element

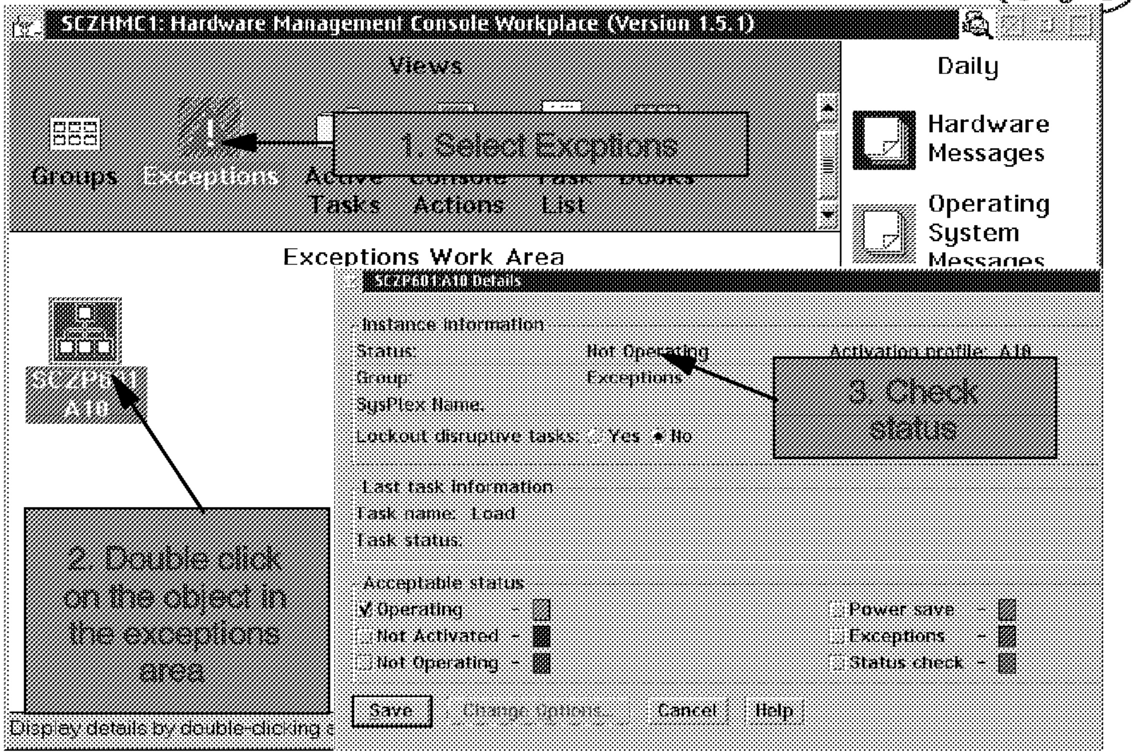
Depending on whether the Lockout disruptive task setting is set to Yes or No determines if you can perform a disruptive task on a CPC or CPC image. You can either lock an individual object or a group of objects at one time. To individually lock a CPC or CPC image:

1. Locate the object you want to lock in the Work Area.
2. Double click on the object′s icon to open its Details page.

3. Set the Lockout disruptive tasks radio button to Yes.

4. Select the Save push button to lock the object.

How to View Exceptions



SCZHMC1: Hardware Management Console Workplace (Version 1.5.1)

SCZP601:A10 Details

Instance information

Status: Not Operating

Activation profile: A10

Group: Exceptions

SysPlex Name:

Lockout disruptive tasks: ○ Yes ● No

Last task information

Task name: Load

Task status:

Acceptable status

☑ Operating - ▨ □ Power save - ▨

□ Not Activated - ▨ □ Exceptions - ▨

□ Not Operating - ▨ □ Status check - ▨

Figure 116. How to view exceptions

6.16 How to view exceptions

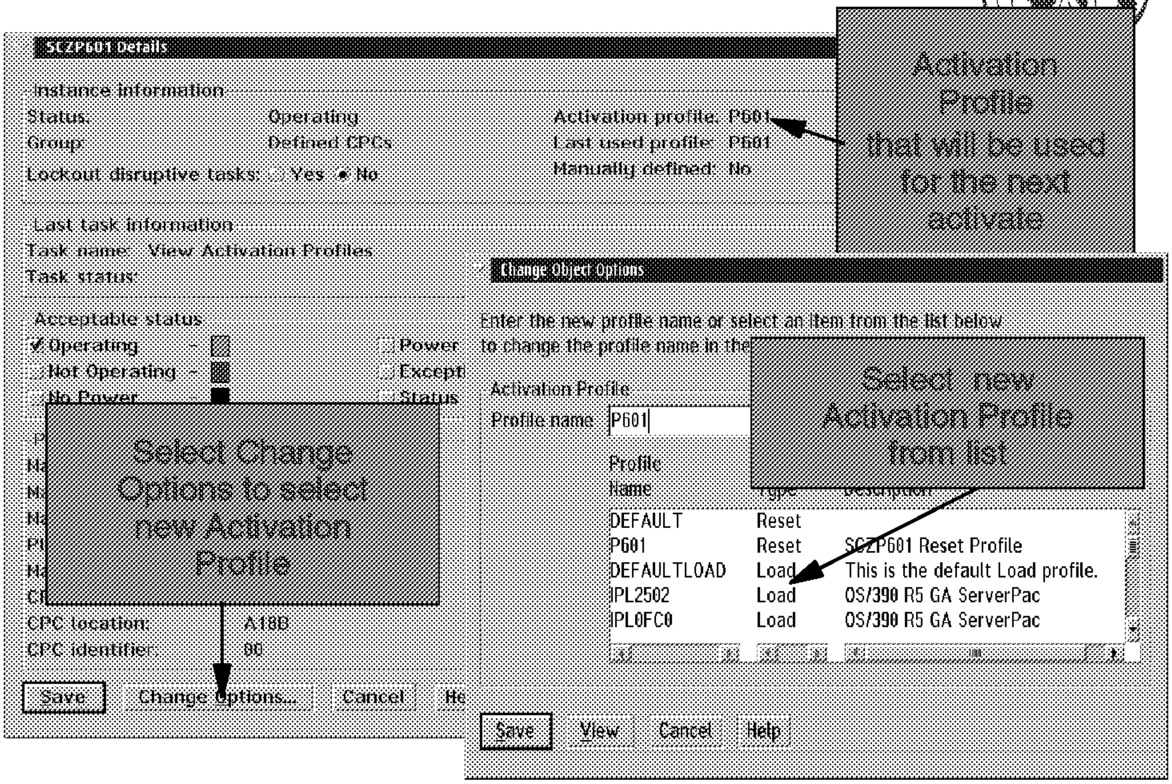
The views area background turns red when an unacceptable status condition exists.

To find the cause of the exception:

1. Double click on the Exceptions View (the view will turn green but the exception icon will stay red).
2. Double click on the object in the Exception work area to display the details panel.
3. Current status is indicated in the upper left corner of the details panel. In this case the image is not operating (image has been system reset).

The color of object in the exceptions area also indicates the type of exception.

Select a New Activation Profile



SCZP601 Details

Instance information

Status: Operating

Group: Defined CPCs

Lockout disruptive tasks: ○ Yes ● No

Activation profile: P601

Last used profile: P601

Manually defined: No

Last task information

Task name: View Activation Profiles

Task status:

Acceptable status

☑ Operating

□ Not Operating

□ No Power

CPC location: A18B

CPC identifier: 00

Change Object Options

Enter the new profile name or select an item from the list below to change the profile name in the

Activation Profile

Profile name: P601

Profile Name	Type	Description
DEFAULT	Reset	
P601	Reset	SCZP601 Reset Profile
DEFAULTLOAD	Load	This is the default Load profile.
IPL2502	Load	OS/390 R5 GA ServerPac
IPL0FC0	Load	OS/390 R5 GA ServerPac

Activation Profile that will be used for the next activate

Select Change Options to select new Activation Profile

Select new Activation Profile from list

Figure 117. Determining or changing the assigned profile

6.17 Determining or changing the assigned profile

To determine the contents of or change the contents of the assigned profile:

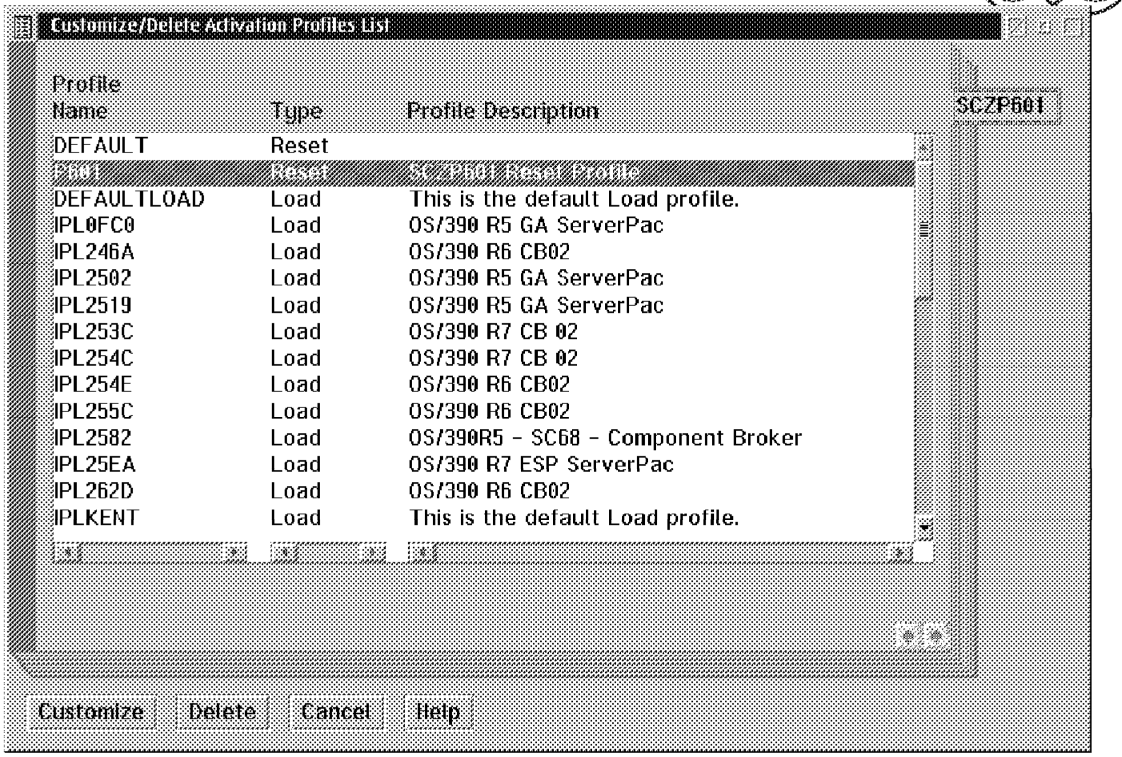
1. Open the groups View.
2. Open the correct Group.
3. Open the object in the Group to display the details panel.

The name of the currently assigned profile is displayed in the upper right-hand corner of the details panel.

If the wrong profile is assigned, click on the change options push-button to allow changing the assigned profile. The change options push-button will display all available profiles stored on the SE for this type of object (CPC or Image). Select the correct profile and then click on Save to assign this profile to this object in this group.

You can also select View from the change object options panel. This will let you view the contents of the profile.

Customize Activation Profile



Customize/Delete Activation Profiles List

SCZP601

Profile Name	Type	Profile Description
DEFAULT	Reset	
P601	Reset	SCZP601 Reset Profile
DEFAULTLOAD	Load	This is the default Load profile.
IPL0FC0	Load	OS/390 R5 GA ServerPac
IPL246A	Load	OS/390 R6 CB02
IPL2502	Load	OS/390 R5 GA ServerPac
IPL2519	Load	OS/390 R5 GA ServerPac
IPL253C	Load	OS/390 R7 CB 02
IPL254C	Load	OS/390 R7 CB 02
IPL254E	Load	OS/390 R6 CB02
IPL255C	Load	OS/390 R6 CB02
IPL2582	Load	OS/390R5 - SC68 - Component Broker
IPL25EA	Load	OS/390 R7 ESP ServerPac
IPL262D	Load	OS/390 R6 CB02
IPLKENT	Load	This is the default Load profile.

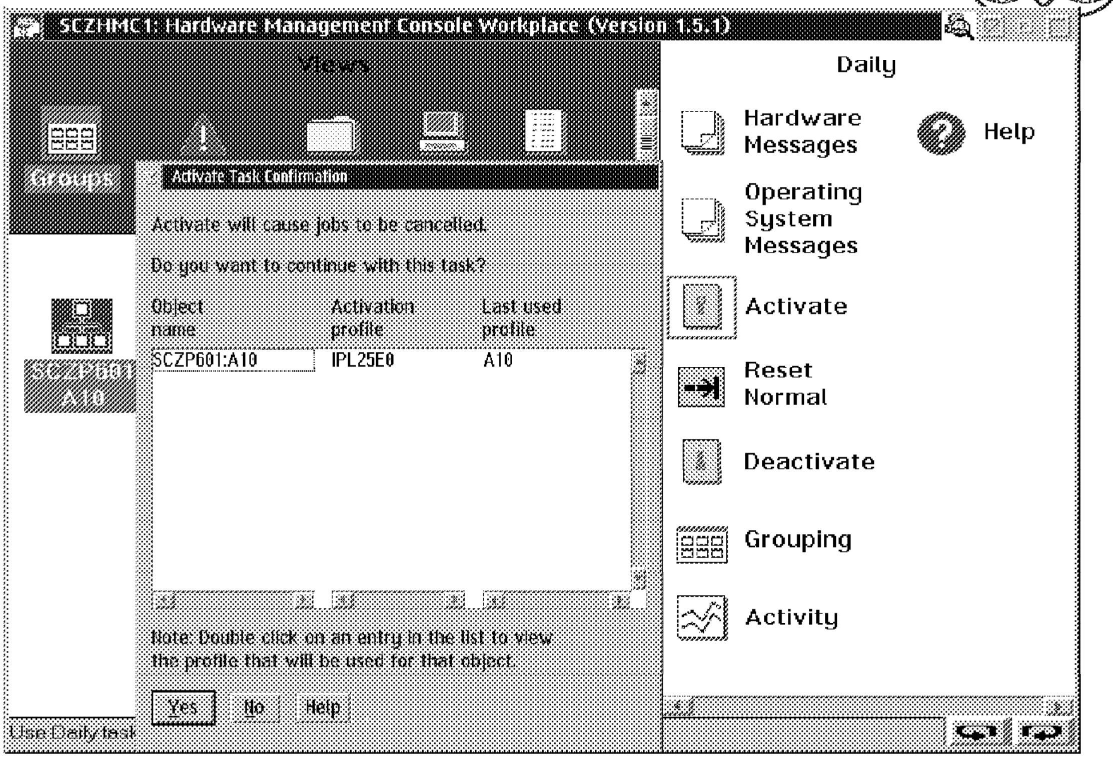
Figure 118. How to create a new activation profile

6.18 How to create a new activation profile

To create a new activation profile:

1. Logon to HMC with System Programmer user ID.
2. Select CPC Operation Customization in the task area.
3. Open the correct group for either a CPC or Image group:
   - A Defined CPC group is used to build new Reset, Image, and Load profiles.
   - A CPC Images group is used to build new Image and Load profiles.
4. Drag and drop the object on Customize/Delete Activation Profile.
5. Select a profile to change or use as a template to create a new profile.
6. Enter a new name for your profile (if creating a new profile).
7. Change the options as required, use the online help to guide your way through the panels as necessary.
8. Select Save, this will save your new profile under the name you previously specified.

How to IPL with a Profile



SCZHMC1: Hardware Management Console Workplace (Version 1.5.1)

Activate Task Confirmation

Activate will cause jobs to be cancelled.

Do you want to continue with this task?

Object name	Activation profile	Last used profile
SCZP601:A10	IPL25E0	A10

Note: Double click on an entry in the list to view the profile that will be used for that object.

Figure 119. How to IPL with a profile

6.19 How to IPL with a profile

First check that you have the correct profile assigned:

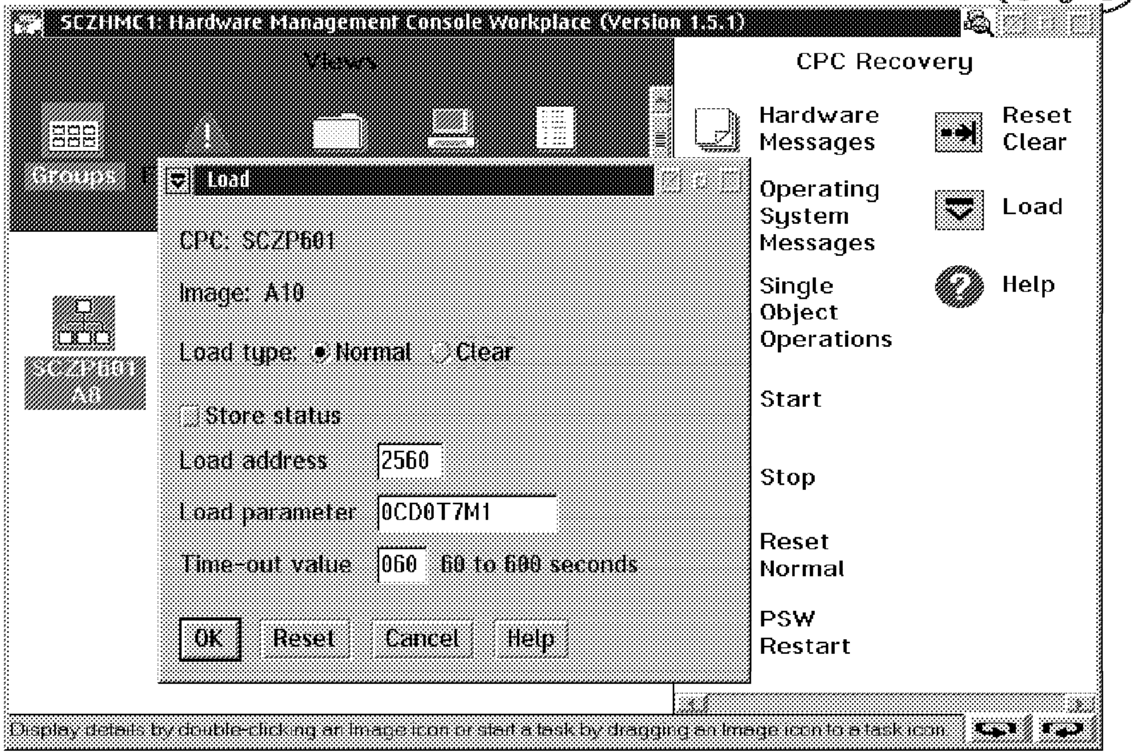
1. Open the groups view.
2. Open the correct group.
3. Open the object in the group to display the Details panel.
4. The name of the currently assigned profile is displayed in the upper right-hand corner of the Details panel. If the wrong profile is assigned click on the Change Options push-button to allow changing the assigned profile.
5. Select the correct profile and select Save to save this profile to this group.

6.19.1 Activate task

The Activate task executes the profile assigned to the object (CPC or Image).

1. Drag and drop the object or objects (if you require multiple activates) on the Activate task.
2. Read the Confirmation panel.
3. Click on Yes if the correct object and profile are assigned.

How to IPL without a Profile



SCZHMC1: Hardware Management Console Workplace (Version 1.5.1)

Load

CPC: SCZP601

Image: A10

Load type: ● Normal ○ Clear

□ Store status

Load address 2560

Load parameter 0CD0T7M1

Time-out value 060 60 to 600 seconds

Display details by double-clicking an image icon or start a task by dragging an image icon to a task icon.

Figure 120. How to IPL without a profile

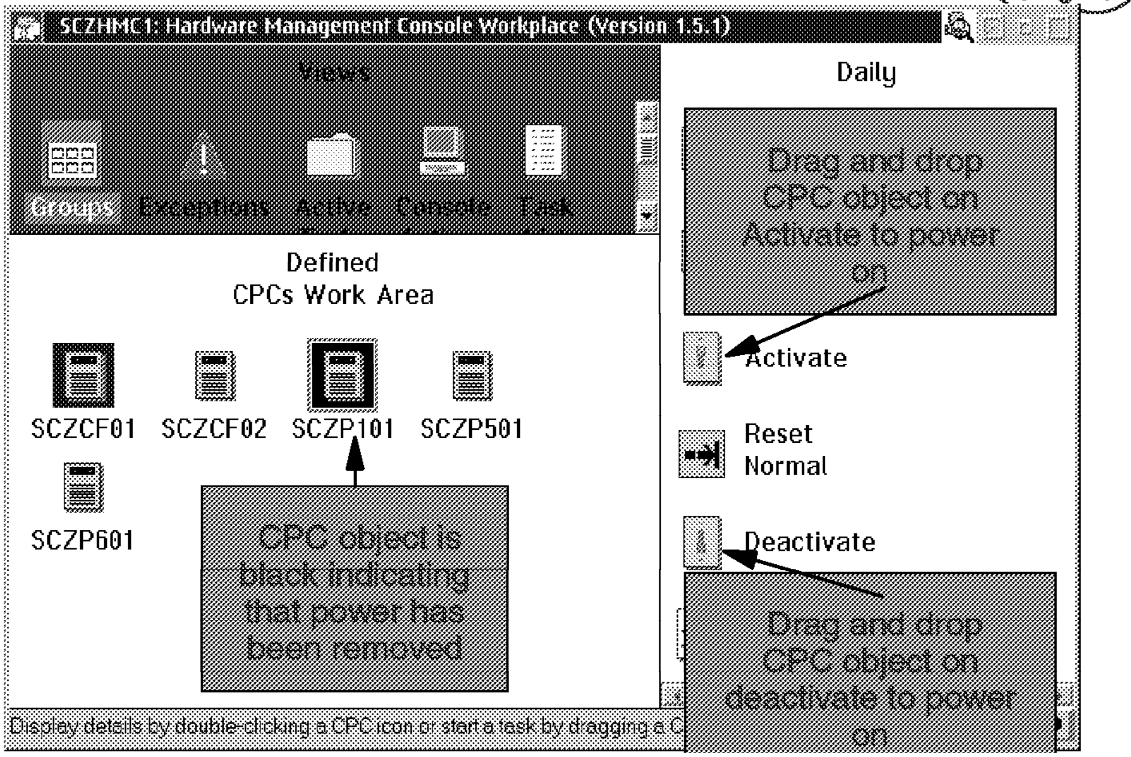
6.20 How to IPL without a profile

To IPL without a profile:

1. Open the Groups view.
2. Select CPC Recovery Task list in the task area.
3. Open the correct images group.
4. Drag and drop the object on the Load task.
5. Enter the Load address.
6. Enter the Load parameters.
7. Select load type Normal or Clear.
8. Click on OK.

The Image will be IPLed with the parameters just entered.

How to Power Off/On a CPC



Drag and drop CPC object on Activate to power on

CPC object is black indicating that power has been removed

Drag and drop CPC object on deactivate to power on

Display details by double-clicking a CPC icon or start a task by dragging a C

Figure 121. How to power off/on a CPC

6.21 How to power off/on a CPC

To power off a CPC, do the following steps:

1. Open the Groups view.
2. Select Daily in the Task area.
3. Open the correct CPC Group.
4. Drag and drop the CPC object that you wish to power off on the Deactivate task.

As the power on also performs a power-on reset (POR), you need to select the correct profile before Activating the CPC.

To check that you have the correct reset profile assigned, do the following:

1. Open the Groups view.
2. Open the correct Group.
3. Open the object in the Group to display the details panel.

The name of the currently assigned profile is displayed in the upper right-hand corner of the Details panel.

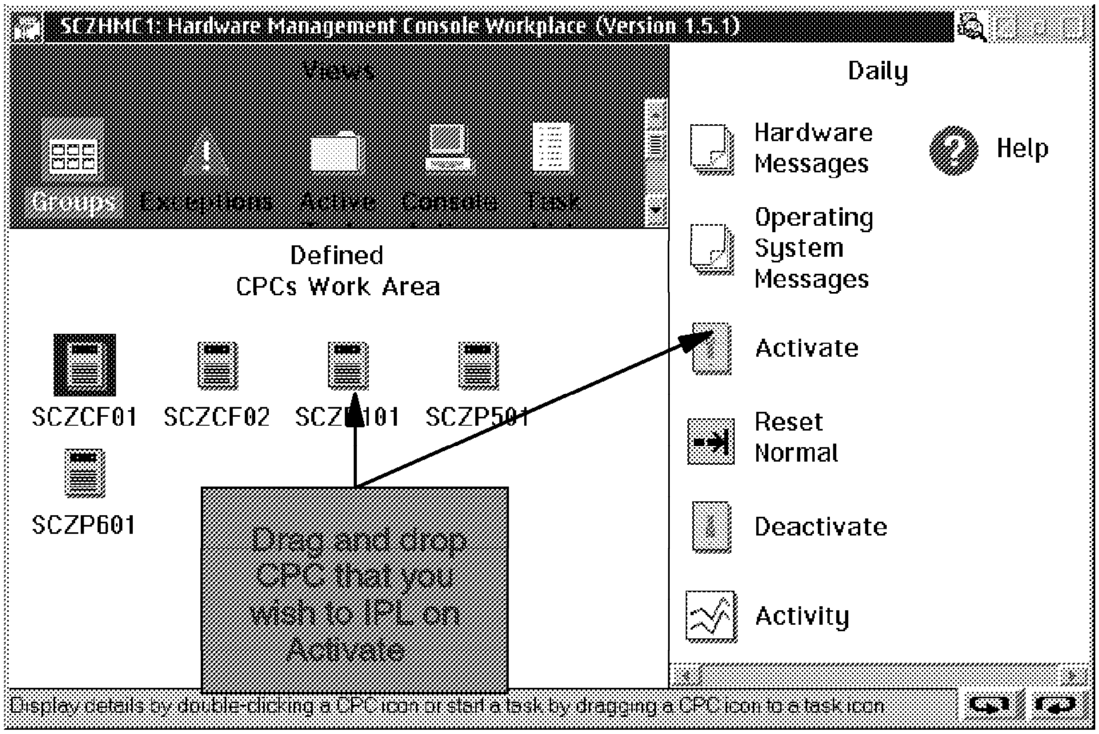
If the wrong profile is assigned, click on the Change Options push-button to allow changing the assigned profile. The Change Options push-button will display all available profiles stored on the SE.

Select the correct profile and then click on Save to assign this profile to the CPC.

You can also select View from the Change Object Options panel. This will let you view the contents of the profile.

Drag and drop the CPC object on the Activate task to Power on the CPC.

How to Power on Reset a CPC



Drag and drop CPC that you wish to IPL on Activate

Display details by double-clicking a CPC icon or start a task by dragging a CPC icon to a task icon

Figure 122. How to power-on reset a CPC

6.22 How to power-on reset a CPC

First, check the Reset profile, then determine that the assigned profile is correct by doing the following:

1. Open the Groups view.
2. Open the correct CPC Group.
3. Open the CPC object in the Group to display the Details panel.

The name of the currently assigned profile is displayed in the upper right-hand corner of the Details panel.

If the wrong profile is assigned, click on the Change Options push-button to allow changing the assigned profile. The Change Options push-button will display all available profiles stored on the SE for this type of object (CPC or Image). Select the correct profile and then click on Save to assign this profile to this object in this group.

6.22.1 How to power-on reset (POR)

Drag and drop the CPC object on the Activate task to POR the CPC.

Note: There is *no* requirement to deactivate the CPC prior to the activate.

Chapter 7. Performance management

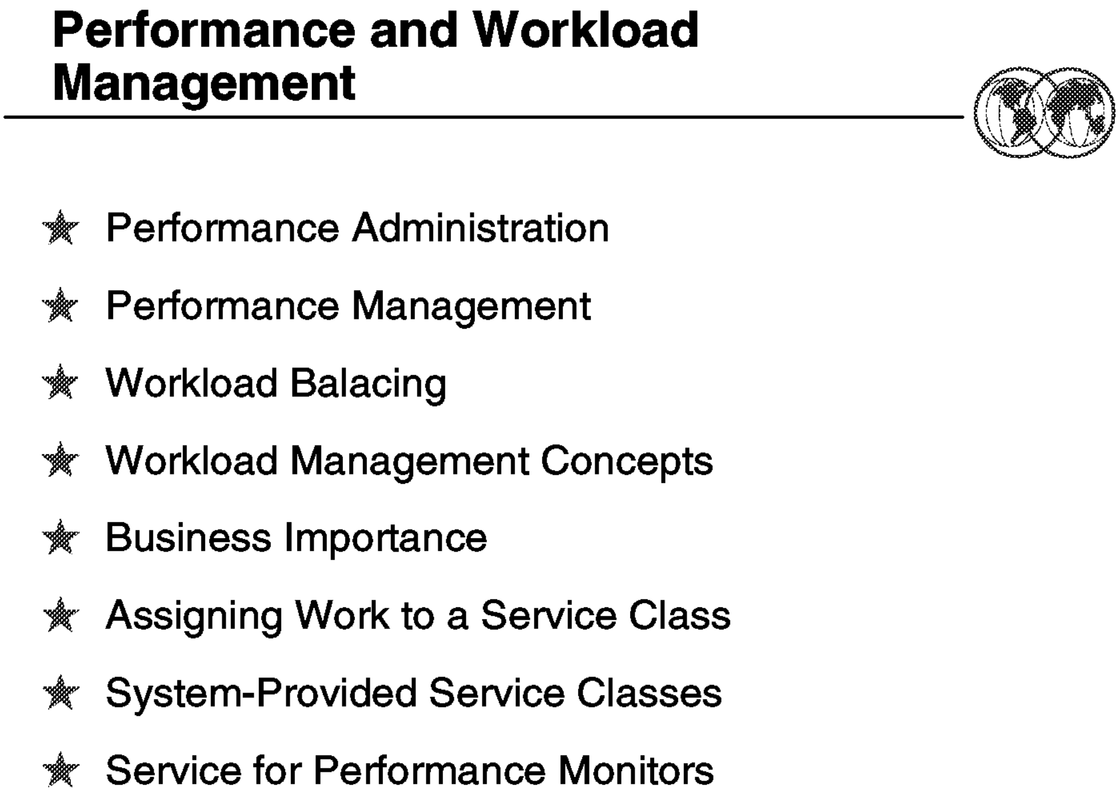
Workload management provides a way to define MVS externals and tune MVS without having to specify low-level parameters. The focus is on setting performance goals for work, and letting the Workload Manager handle processing to meet the goals. The IPS and ICS parmlib members, as well as certain parts of the OPT parmlib member, need no longer to be used to specify resource management.

Workload management provides new MVS performance management externals in a service policy that reflects goals for work, expressed in terms commonly used in service level agreements (SLA). Because the terms are similar to those commonly used in an SLA, you can communicate with end users, with business partners, and with MVS using the same terminology.

With one common terminology, workload management provides feedback to support performance reporting, analysis, and modelling tools. The feedback describes performance achievements in the same terms as those used to express goals for work.

Workload management provides automatic work and resource management support that dynamically adapts as needed. It manages the trade-offs between meeting your service goals for work and making efficient use of system resources.

Workload management eliminates the need to micro-manage each individual MVS image, providing a way for you to increase the number of systems in your installation without greatly increasing the necessary skill level.



Performance and Workload Management

- Performance Administration
- Performance Management
- Workload Balacing
- Workload Management Concepts
- Business Importance
- Assigning Work to a Service Class
- System-Provided Service Classes
- Service for Performance Monitors

Figure 123. Performance and workload management

7.1 What is OS/390 performance and workload management

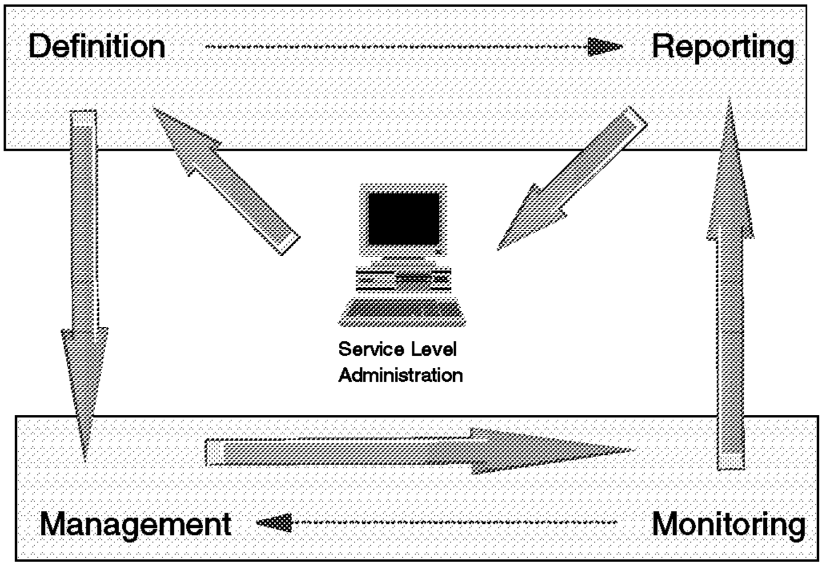
Installations today process different types of work with different response times. Every installation wants to make the best use of its resources and maintain the highest possible throughput and achieve the best possible system responsiveness.

With workload management, you define performance goals and assign a business importance to each goal. You define the goals for work in business terms, and the system decides how much resource, such as CPU and storage, should be given to the work to meet its goal.

An installation should know what it expects to accomplish in the form of performance goals, as well as how important it is to the business that each performance goal be achieved. With workload management, you define performance goals for work, and the system matches resources to the work to meet those goals, constantly monitoring and adapting processing to meet the goals. Reporting reflects how well the system is doing compared to its goals.

Workload Management Overview

Performance Administration



Performance Management

Figure 124. Workload management overview

7.1.1 Performance administration

Performance administration is the process of defining and adjusting performance goals. Workload management introduces the role of the service level administrator. The service level administrator is responsible for defining the installation′s performance goals based on business needs and current performance. This explicit definition of workloads and performance goals is called a service definition. Some installations might already have this kind of information in a service level agreement (SLA). The service definition applies to all types of work, CICS, IMS, TSO/E, OS/390 UNIX System Services, JES, APPC/MVS, LSFM, DDF, DB2, SOM, Internet Connection Server and others. You can specify goals for all OS/390-managed work, whether it is online transactions or batch jobs. The goals defined in the service definition apply to all work in the sysplex.

Because the service definition terminology is similar to the terminology found in an SLA, the service level administrator can communicate with the installation user community, with upper level management, and with OS/390 using the same terminology. When the service level requirements change, the service level administrator can adjust the corresponding workload management terms, without having to convert them into low-level OS/390 parameters.

Workload management provides an online panel-based application for setting up and adjusting the service definition. You specify the service definition through this ISPF administrative application.

Workload management provides the capability to collect performance and delay data in context of the service definition. The performance and delay data are available to reporting and monitoring products, so that they can use the same terminology.

7.1.2 Performance management

Performance management is the process workload management uses to decide how to match resources to work according to performance goals. Workload management algorithms use the service definition information and internal monitoring feedback to check how well they are doing in meeting the goals. The algorithms periodically adjust the allocation of resource as the workload level changes.

For each system, workload management handles the system resources. Workload management coordinates and shares performance information across the sysplex. How well it manages one system is based on how well the other systems are also doing in meeting the goals. If there is contention for resources, workload management makes the appropriate trade-offs based on the importance of the work and how well the goals are being met.

Workload management can dynamically start and stop server address spaces to process work from application environments. Workload management starts and stops server address spaces in a single system or across the sysplex to meet the work′s goals.

You can turn over management of batch initiators to workload management, allowing workload management to dynamically manage the number of batch initiators for one or more job classes to meet the performance goals of the work.

In addition to internal feedback monitoring, workload management keeps track of what is happening in the sysplex in the form of real time performance data collection, and delay monitoring. All this information ids available for performance monitors and reporters for integration into detailed reports.

Workload Balancing

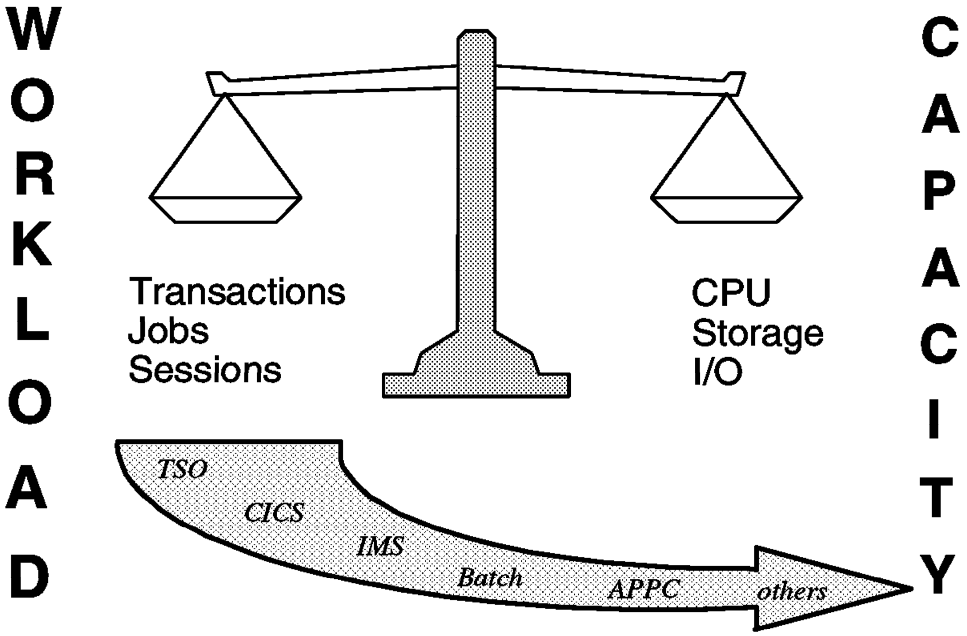


Figure 125. Workload balancing

7.1.3 Workload balancing

To make the most of workload management, work needs to be properly distributed so that OS/390 can manage the resources. It is essential that the subsystems distributing work are configured properly for workload distribution in a sysplex. You do this with the controls provided by each subsystem. For example, in a JES2 and JES3 environment, you spread initiator address spaces across each system.

Initial cooperation between OS/390 and the transaction managers (CICS, IMS, DB2) allows you to define performance goals for all types of OS/390-managed work. Workload management dynamically matches resources (access to the processor and storage) to work to meet the goals.

CICS, however, goes further with the CICSplex Systems Manager (CICSPlex SM) to dynamically route CICS transactions to meet the performance goals. CPSM monitors the performance of CICS resources and systems and presents the resources as though they were part of a single system. This type of cooperation greatly improves CICS transaction workload balancing.

Other subsystems also have automatic and dynamic work balancing in a sysplex. For example, DB2 can spread distributed data facility (DDF) work across a sysplex automatically. DB2 can also distribute work in a sysplex through its sysplex query parallelism function. CICS, TSO, and APPC cooperate with VTAM and workload management in a sysplex to balance the placement of sessions. SOMobjects can automatically spread its servers across a sysplex to meet performance goals and to balance the work.

Workload Management Concepts

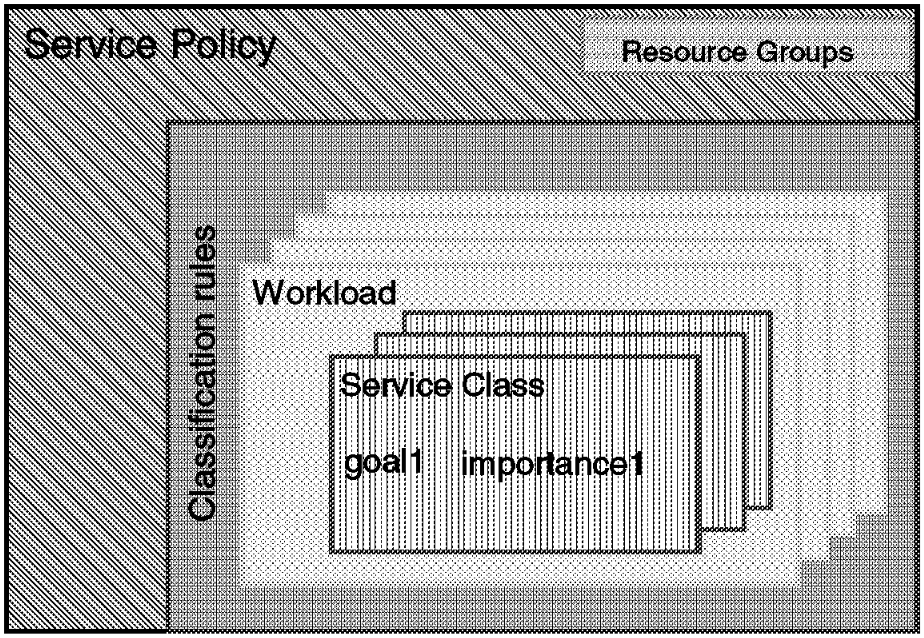


Figure 126. Workload Manager concepts

7.1.4 Workload Manager concepts

The service definition contains all the information about the installation needed for workload management processing. There is one service definition for the entire sysplex. The service level administrator specifies the service definition through the WLM administrative application. The service level administrator sets up "policies" within the service definition to specify the goals for work. A service level administrator must understand how to organize work, and be able to assign it performance objectives.

A service definition consists of:

- Service policies
- Workloads
- Service classes
- Report classes
- Resource group
- Classification rules
- Application environments
- Scheduling environments

To workload management, work is a demand for service, such as a batch job, an APPC, CICS, DB2, or IMS transaction, a TSO/E logon, a TSO/E command, or a SOM request. All work running in the installation is divided into workloads. A *workload* is a group of work that is meaningful for an installation to monitor. For example, all the work created by a development group could be a workload, or all the work started by an application, or in a subsystem.

Within a workload, you group work with similar performance characteristics into service classes. You create a service class for a group of work with similar performance goals, resource requirements and business importance to the installation.

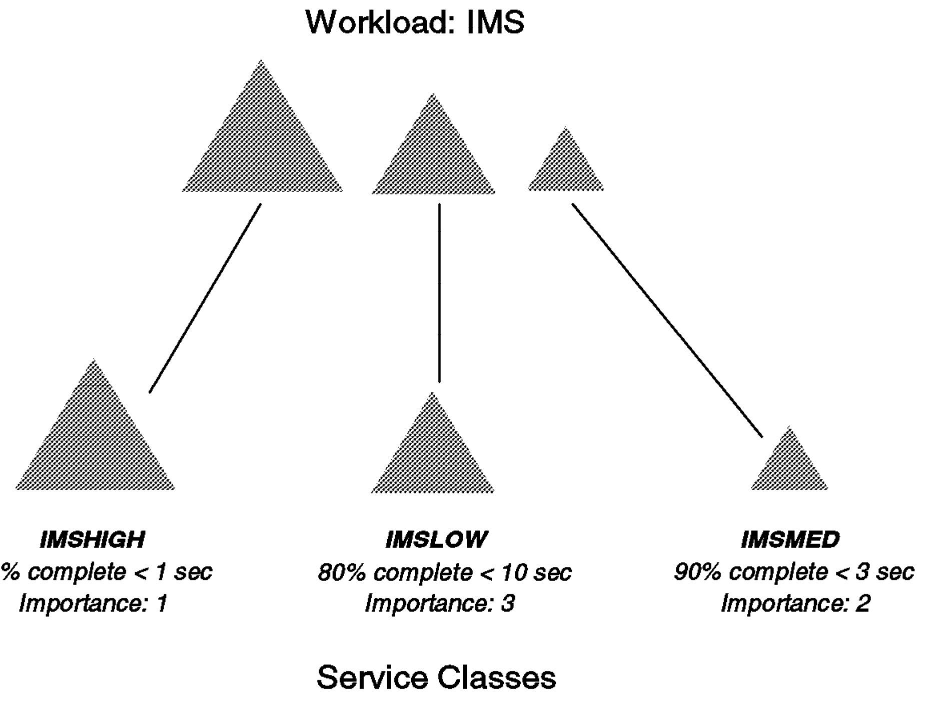
7.1.4.1 Performance goals

You assign a performance goal to each service class period, such as a response time goal, and you indicate an importance. Importance is how important it is to your business that the performance goal be achieved.

There are three kinds of goals: response-time, execution velocity, and discretionary. Response time goals indicate how quickly you want your work to be processed. Since response time goals are not appropriate for all kinds of work, such as long-running batch jobs, there are execution velocity goals.

Execution velocity goals define how fast work should run when ready, without being delayed for processor, storage, I/O access, and queue delay. Execution velocity goals are intended for work for which response time goals are not appropriate, such as started tasks, or long running batch work. Discretionary goals are for low priority work for which you do not have any particular performance goal. Workload management then processes the work using resources not required to meet the goals of other service classes.

Workload Organized by Subsystem



Workload: IMS

IMSHIGH
95% complete < 1 sec
Importance: 1

IMSLOW
80% complete < 10 sec
Importance: 3

IMSMED
90% complete < 3 sec
Importance: 2

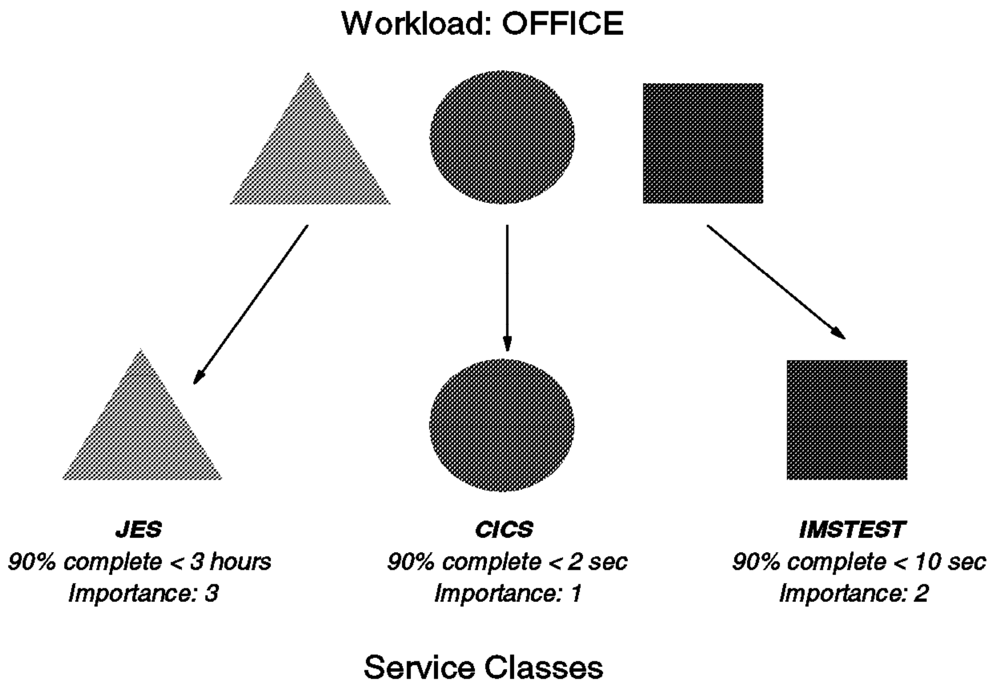
Service Classes

Figure 127. Workload organized by subsystem

7.1.5 Business importance

When there is not sufficient capacity for all work in the system to meet its goals, business importance is used to determine which work should give up resources and which work should receive more. You assign an importance to a service class period, which indicates how important it is that the goal be met relative to other goals. Importance plays a role only when a service class period is not meeting its goal. There are five levels of importance: lowest (5), low, medium, high, and highest (1). The visual shows an example of the relationship between workloads and service classes. Work in the figure is represented by different size triangles. The IMS workload in the figure represents all of the IMS work. There are three single-period service classes set up, each with a different importance, and a different response time goal.

Workload Organized by Department



Workload: OFFICE

JES
90% complete < 3 hours
Importance: 3

CICS
90% complete < 2 sec
Importance: 1

IMSTEST
90% complete < 10 sec
Importance: 2

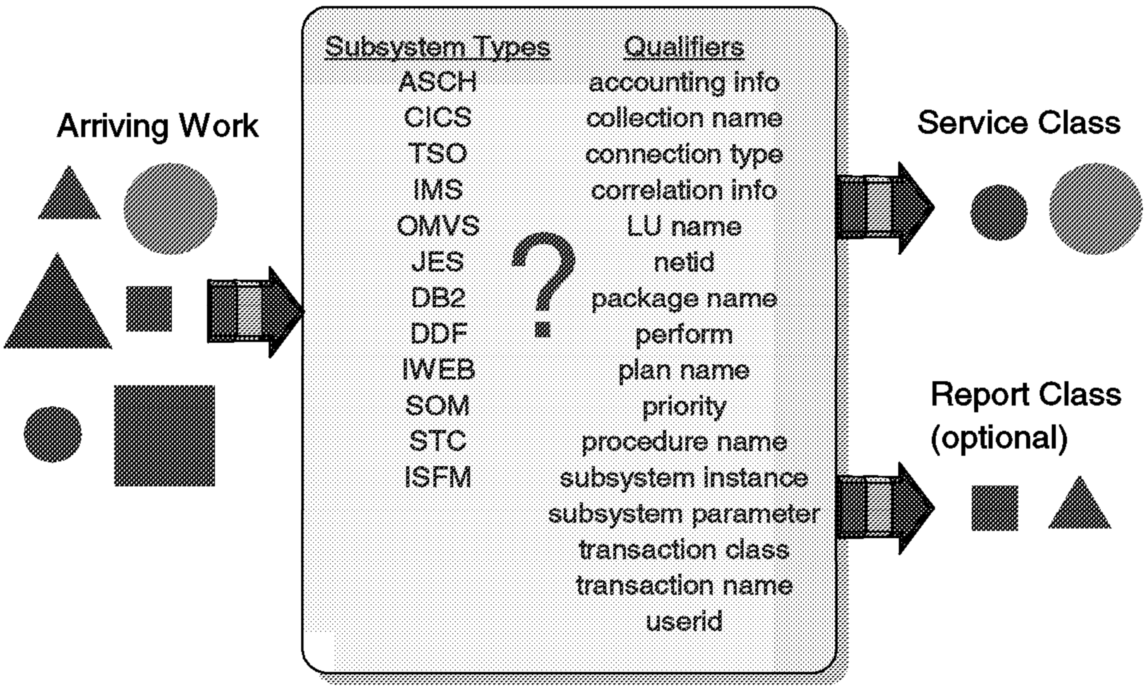
Service Classes

Figure 128. Workload organized by department

7.1.6 Workload organized by department

The visual shows an example of a workload that is organized by division. In the figure, work is represented by different shapes: circles, squares, and triangles. The workload, OFFICE, represents all of the work in the office division. There are three service classes set up, each a different kind of work in the OFFICE workload. IMSTEST represents the IMS test work, CICS represents all of the CICS work, and JES represents all of the batch work in the OFFICE workload. Each of the service classes has one period with a different response time goal assigned to it.

Work Classification



Subsystem Classification Rules

Subsystem Types

ASCH
CICS
TSO
IMS
OMVS
JES
DB2
DDF
IWEB
SOM
STC
ISFM

Qualifiers

accounting info
collection name
connection type
correlation info
LU name
netid
package name
perform
plan name
priority
procedure name
subsystem instance
subsystem parameter
transaction class
transaction name
userid

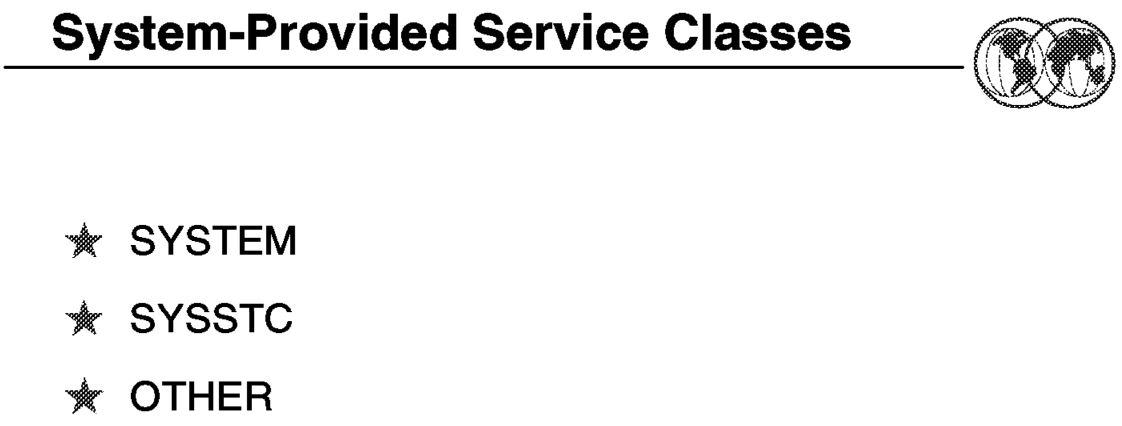
Figure 129. Workload classification

7.1.7 Assigning work to a service class

Classification rules are the rules that workload management uses to associate a performance goal and/or a resource group with work by associating incoming work with a service class. Optionally, classification rules can also associate incoming work with a report class, similar to a report performance group. Classification rules work in much the same way as the IEAICSxx parmlib member in compatibility mode.

The classification rules for a subsystem are specified in terms of transaction qualifiers such as job name or transaction class. These qualifiers identify groups of transactions that should have the same performance goals and importance. The attributes of incoming work are compared to these qualifiers and, if there is a match, the rule is used to assign a service class to the work. A subsystem can also have a default class for work that does not match any of the rules.

Optionally, classification rules can assign incoming work to a report class. You get reporting information in terms of workloads, service classes, and report classes. Use report classes for obtaining data on a subset of work within a service class, or for rolling up data across workloads.



System-Provided Service Classes

- SYSTEM
- SYSSTC
- OTHER

Figure 130. System-provided service classes

7.1.8 System-provided service classes

If some work comes into a system for which there is no associated service class defined in the classification rules, workload management assigns it to a default service class. There are three such default service classes:

SYSTEM For all system address spaces designated “high dispatching priority” (X′FF′) address spaces. The high dispatching priority address spaces include: MASTER, GRS, DUMPSRV, SMF, CATALOG, RASP, XCFAS, SMXC, CONSOLE, IOSAS, and others. For a list of the high dispatching priority address spaces in your installation, see the RMF Monitor II report and look for the x′FF′ dispatching priority.

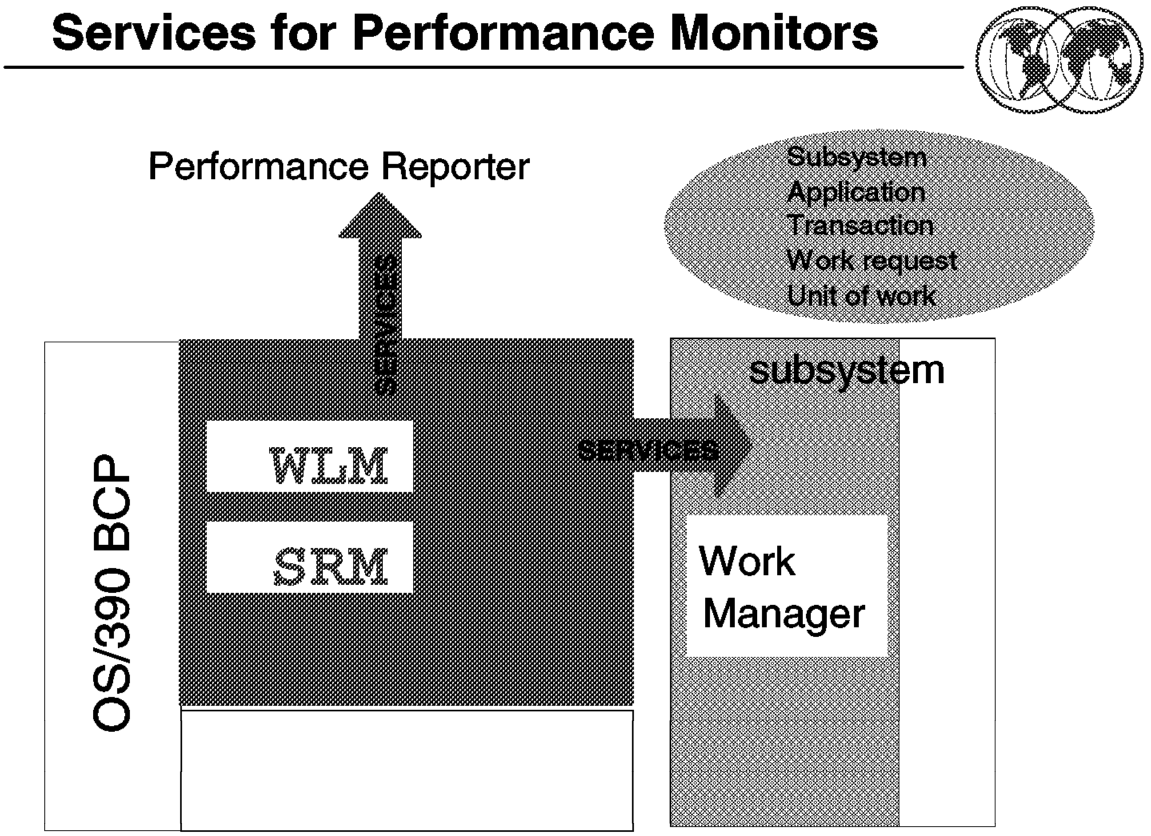
You do not need to set up service classes for these system address spaces. Workload management recognizes these as special system address spaces and treats them accordingly.

SYSSTC For all started tasks not otherwise associated with a service class. Workload management treats work in SYSSTC just below special system address spaces in terms of dispatching.

You can also assign work to the SYSSTC service class as part of your work classification rules. You can do this for classification rules in the subsystem types ASCH, JES, OMVS (OS/390 UNIX System Services), STC and TSO.

Some address spaces normally created when running OS/390 are neither high dispatching priority, privileged, nor a system task, such as NETVIEW. These address spaces must be explicitly assigned to a service class such as SYSSTC.

OTHER For all other work not associated with a service class. This is intended as a "catcher" for all work whose subsystem type has no classification. It is assigned a discretionary goal.



Services for Performance Monitors

Figure 131. Services for performance monitors

7.1.9 Services for performance monitors

The workload reporting services are intended for use by monitoring or reporting products to collect performance data. These services replace some of the existing methods of collecting data, and provide as complete a picture of performance information as possible.

A workload management ISPF application contains an installation′s goals for work in a service policy. The reporting services access the service policy information, and report on how well the installation is doing in processing towards the goals in the policy. The services report information based on the service classes defined in the service policy. They also provide delay information on work managed by subsystems using the execution delay monitoring services.

Because the system collects performance data continually, there is no set reporting interval. So, unlike earlier releases of OS/390, multiple performance monitors can request the services at the same time. And, performance monitors can collect the data based on their own reporting intervals. When the performance monitor invokes a service to collect performance data, the data is provided in a cumulative fashion.

When a significant change occurs in workload management, such as a change from goal mode to compatibility mode, or a policy activation, the data collection is stopped and re-started. At such times, performance monitors should also stop and re-start their reporting intervals. For each time that data collection is stopped and re-started in workload management, an ENF signal notifies listeners of the change.

The workload reporting services provide information for performance monitors to report on how well an installation is doing in meeting performance goals.

You can use the workload reporting services regardless of whether a system is running in goal mode with an active service policy, or whether it is running in compatibility mode with the IEAICSxx and IEAIPSxx parmlib members. While the services apply in both goal mode and compatibility mode, some of the collected data is different for each. The performance monitor must be aware which mode a system is running in, and be able to locate and use the collected performance data appropriately.

For goal mode, with the cooperation of subsystem work managers, the service can provide more performance data than previously reported. They provide information about work that is processed by many address spaces, and allow for a view of subsystem transactions, not just address spaces and enclaves. The data includes response time information, response time distributions, execution delay state information for transactions and information about service classes that different address spaces are serving.

The services allow a performance monitor to show the goal for a service class period, how well the system is doing to meet the goal, and if it is not meeting the goal, why it is delayed. The performance monitor can show this goal vs actual data in terms that are consistent for all OS/390-managed work.

System Resources Manager (SRM)

- System tuning and SRM
- Controlling SRM and objectives
- Types of control
- Pre-installation requirements
- Operator commands related to SRM

Figure 132. System Resources Manager (SRM)

7.2 Performance and System Resources Manager (SRM)

An installation′s control over the performance of the system is exercised through the System Resources Manager (SRM).

The SRM is OS/390 system code running in the nucleus without an address space of its own, and a WLM address space is activated at IPL. The SRM belongs to a set of functions called workload management. The SRM can run in two modes:

- Compatibility mode

  In compatibility mode, all the parameters defined in the OPT, ICS, and IPS members are valid with the exception of APGRNG; APGRNG always defaults to (0,15).

- Goal mode

  In goal mode, the SRM algorithms are driven by WLM using the installation goals and not by the former IPS and ICS PARMLIB constructs.

System Tuning and SRM

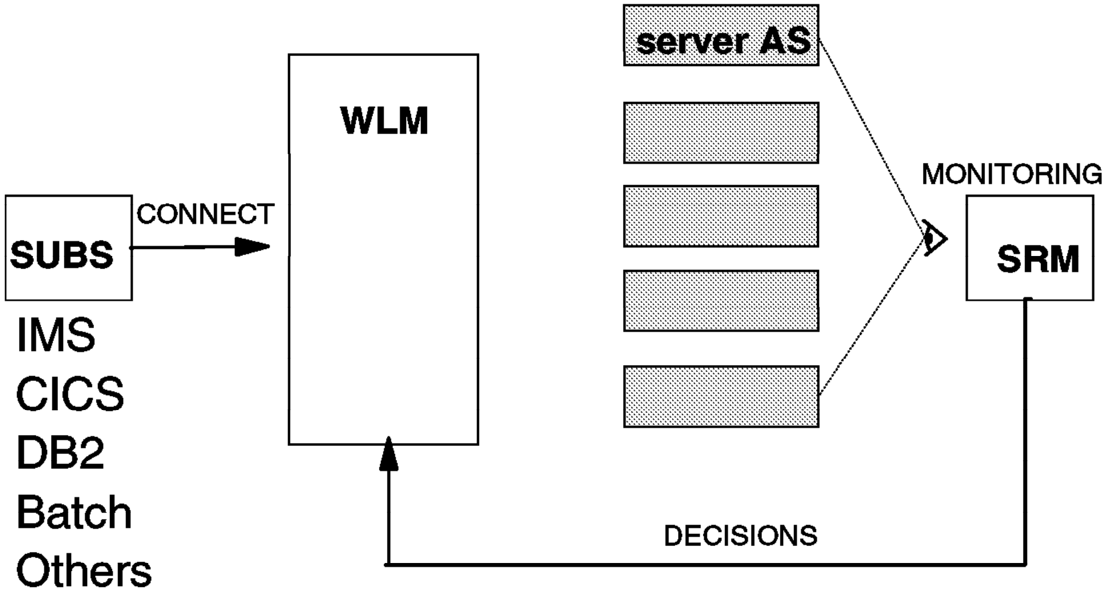


Figure 133. System tuning and SRM

7.2.1 System tuning and SRM

SRM is a component of the system control program (BCP). It determines which address spaces, of all active address spaces, should be given access to system resources and the rate at which each address space is allowed to consume these resources.

Before an installation turns to SRM, it should be aware of the response time and throughput requirements for the various types of work that will be performed on its system. Questions similar to the following should be considered:

- How important is turnaround time for batch work, and are there distinct types of batch work with differing turnaround requirements?
- Should subsystems such as IMS and CICS be controlled at all, or should they receive as much service as they request? That is, should they be allowed unlimited access to resources without regard to the impact this would have on other types of work?
- What is acceptable TSO/E response time for various types of commands?
- What is acceptable response time for compiles, sorts, or other batch-like work executed from a terminal?

Once these questions have been answered and, whenever possible, quantified, and the installation is reasonably confident that its requirements do not exceed the physical capacity of its hardware, it should then turn to SRM to specify the desired degree of control.

The task of tuning a system is an iterative and continuous process. The controls offered by SRM are only one aspect of this process. Initial tuning consists of selecting appropriate parameters for various system components and subsystems. Once the system is operational and criteria have been established for the selection of jobs for execution via job classes and priorities, SRM will control the distribution of available resources according to the parameters specified by the installation.

SRM, however, can only deal with available resources. If these are inadequate to meet the needs of the installation, even optimal distribution may not be the answer--other areas of the system should be examined to determine the possibility of increasing available resources.

When requirements for the system increase and it becomes necessary to shift priorities or acquire additional resources, such as a larger processor, more storage, or more terminals, the SRM parameters might have to be adjusted to reflect changed conditions.

Controlling SRM

SYS1.PARMLIB

★ IEAIPSxx

.....

★ IEAICSxx

.....

★ IEAOPTxx

.....

Figure 134. Controlling SRM

7.2.2 Controlling SRM

You have two options for controlling SRM: through the IEAIPSxx and IEAICSxx parmlib members, or through the Workload Manager. Controlling SRM through parmlib members is called workload management *compatibility mode*, and controlling SRM through the Workload Manager is called *goal mode*. Some parameters in the IEAOPTxx parmlib member apply to both goal and compatibility mode.

Because controlling SRM in compatibility mode can be a complex process, we recommend that you use Workload Manager goal mode. With Workload Manager, you specify performance goals for work, and SRM adapts the system resources to meet the goals. SRM uses the same controls that exist today, but it sets them all dynamically based on the goals.

7.2.2.1 Objectives

SRM bases its decision on two fundamental objectives:

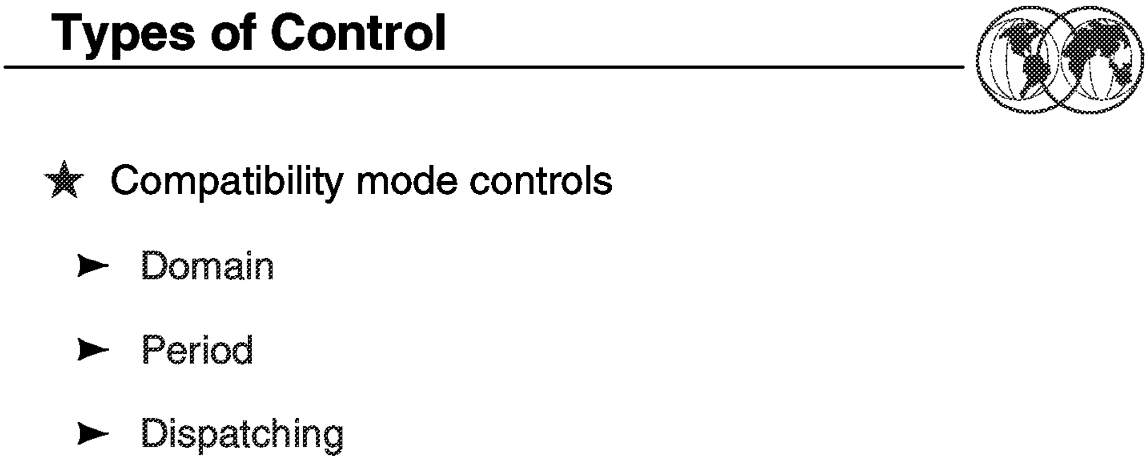
1. To distribute system resources among individual address spaces in accordance with the installation′s response, turnaround, and work priority requirements.
2. To achieve optimal use of system resources as seen from the viewpoint of system throughput.

An installation specifies its requirements in a member of parmlib called the installation performance specification (IPS). Through the IPS, the installation divides its types of work into distinct groups, called domains, assigns relative importance to each domain, and specifies the desired control characteristics for each address space within these domains. The meaning and use of domains is discussed in detail in succeeding sections.

The installation control specification (ICS), another member of parmlib, assigns a set of IPS performance characteristics to each address space. The installation control specification can also be used to tailor the RMF reports for TSO/E, batch, and started tasks. These reports are useful for the continuous evaluation of system performance. RMF can also report on the transactions of subsystems that invoke the SYSEVENT macro.

Another member of parmlib, the OPT member, contains parameters used to balance the use of the system′s resources. Through a combination of IPS, OPT, and ICS parameters, an installation can exercise a degree of control over system throughput characteristics (objective number 2). That is, the installation can specify whether, and under what circumstances, throughput considerations are more important than response and turnaround requirements when the need arises to make trade-offs between objective number 1 and objective number 2.

SRM attempts to ensure optimal use of system resources by periodically monitoring and balancing resource utilization. If resources are underutilized, SRM will attempt to increase the system load. If resources are overutilized, SRM will attempt to reduce the system load.



Types of Control

- Compatibility mode controls
  - Domain
  - Period
  - Dispatching

Figure 135. Types of control

7.2.3 Types of control

SRM offers three distinct types of control to an installation:

Domain Domain control for swappable address spaces

Domains allow an installation to divide its types of work into distinct groups and thereby exercise individually-tailored control over different types of work, such as batch work, IMS message processors, short TSO/E commands and long-running TSO/E commands.

Domains, therefore, are simply the means by which related types of work are grouped together, such that all work (address spaces) assigned to one domain has some common set of characteristics that differentiates it from the work assigned to other domains. What constitutes such a set of characteristics is entirely dependent on an installation′s requirements. Work could be differentiated on the basis of execution characteristics such as short- versus long-running and batch versus foreground, or according to different use characteristics such as IMS message processors and student or test programs. On the other hand, an installation may choose to use different user requirements as the basis for differentiating and divide its user population into domains, regardless of the execution characteristics of individual jobs. A mixture of the above considerations is, of course, equally possible.

Period Period control for all address spaces

Additional flexibility of control is gained by dividing the life span of a transaction (a unit of work that is consuming service) into distinct performance periods, and associating an address space with a different set of performance characteristics for the duration of each period. Performance periods are specified via the IPS.

The purpose of performance periods is to allow an installation to vary the performance characteristics of transactions as their execution characteristics change. For example, if a domain includes a variety of TSO/E users, transactions with greatly differing life spans will be competing with one another. If the majority of transactions are of short duration and these are to experience consistently good response time, it may not be satisfactory to keep one set of performance characteristics in effect for the entire life span of every transaction. By using performance periods, short transactions, for instance, can be favored over long transactions without prior knowledge of how long a transaction runs.

Dispatching Dispatching control for all address spaces.

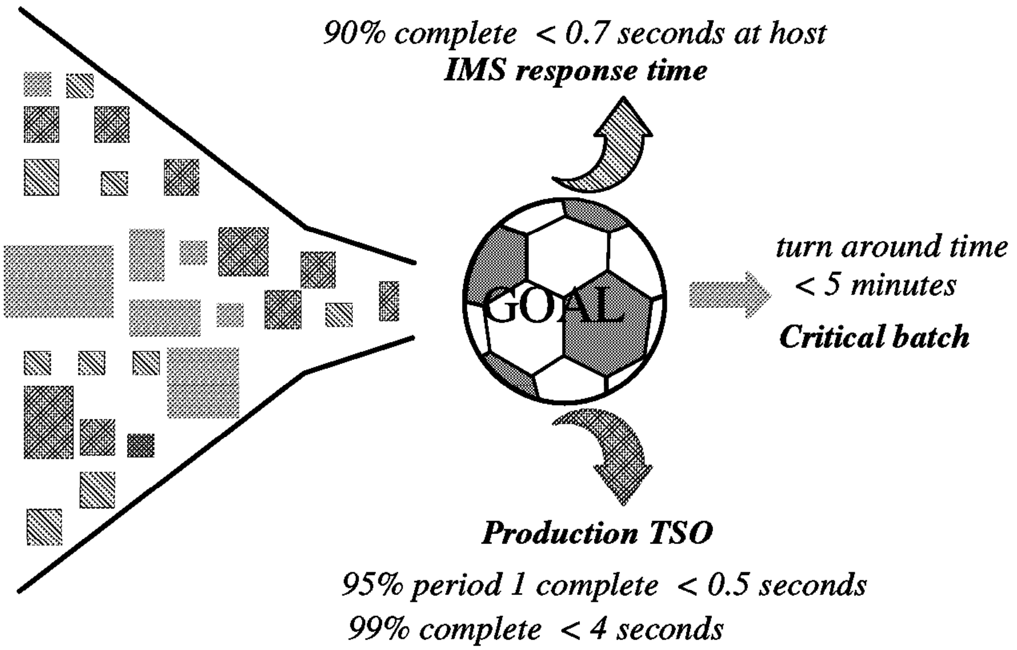
Dispatching priorities control the rate at which address spaces are allowed to consume resources after they have been given access to these resources. This form of competition takes place outside the sphere of domain control, that is, all address spaces compete with all other address spaces with regard to dispatching priorities.

The installation sets dispatching priorities through the IPS, and can alter them as the address space′s execution characteristics change.

The installation control specification, IPS, and OPT contain the parameters used to modify these controls. These are the parameters you use for compatibility mode.

For goal mode, SRM still uses domain and dispatching controls, but it sets the values dynamically based on the performance goals for work defined in a service policy.

Pre-installation Requirements



90% complete < 0.7 seconds at host
IMS response time

turn around time
< 5 minutes
Critical batch

Production TSO
95% period 1 complete < 0.5 seconds
99% complete < 4 seconds

Figure 136. Pre-installation requirements

7.2.4 Pre-installation requirements

Before specifying any parameters to SRM, an installation must define response and throughput requirements for its various classification of work.

Examples of specific questions that should be answered are listed below. The applicability of these questions will, of course, vary from installation to installation.

- **Subsystems**

  How many subsystems will be active at any one time and what are they?

  For IMS, how many active regions will there be?

  Will the subsystem address space(s) be nonswappable?

  What is the desired response time and how will it be measured?

- **Batch**

  What is the required batch throughput or turnaround for various job classes?

  How much service do batch jobs require, and what service rate is needed to meet the turnaround requirement?

  An RMF workload report or reduction of SMF data in type 5 or type 30 records will provide the average service consumed by jobs of different classes. Based on service definition coefficients of CPU=10.0,IOC=5.0,MSO=3.0,SRB=10.0; the following approximations can be made:

  Short jobs use 30,000 service units or less.

Long jobs use more than 30,000 units.

What is the average number of ready jobs?

Most likely, this is the number of active initiators. A few extra initiators may be started to decrease turnaround times.

- **TSO/E**

  What is the number of terminals?

  What is the average number of ready users?

  As a guideline for installations new to TSO/E, assume that an installation doing program development on 3270 terminals will have two ready users for every ten users logged on. This average will vary, depending on the type of terminal and on the type of TSO/E session (data entry, problem solving, program development).

  What is the required response time and expected transaction rate for different categories of TSO/E transactions at different times, such as peak hours?

  What is the expected response time for short transactions?

  How will this response time be measured?

  Should response time be different for select groups of TSO/E users?

  How should semi-trivial and non-trivial transactions be treated?

  How are they defined?

  An installation can use RMF workload reports or SMF data in type 34 and 35 records available to help define trivial and non-trivial TSO/E work. Based on service definition coefficients of CPU=10.0,IOC=5.0,MSO=3.0,SRB=10.0; the following approximations can be made:

  Short TSO/E commands use 200 service units or less.

  Medium length commands use between 200 and 1000 service units.

  Long TSO/E commands use 1000 service units or more.

  What is the required service rate for TSO/E users?

  If 2-second response time (as reported by RMF) is required for very short TSO/E commands (100 service units), the required service rate for such a transaction is 100/2 or 50 service units per second. Service rates for other types of transactions should be computed also.

- **General**

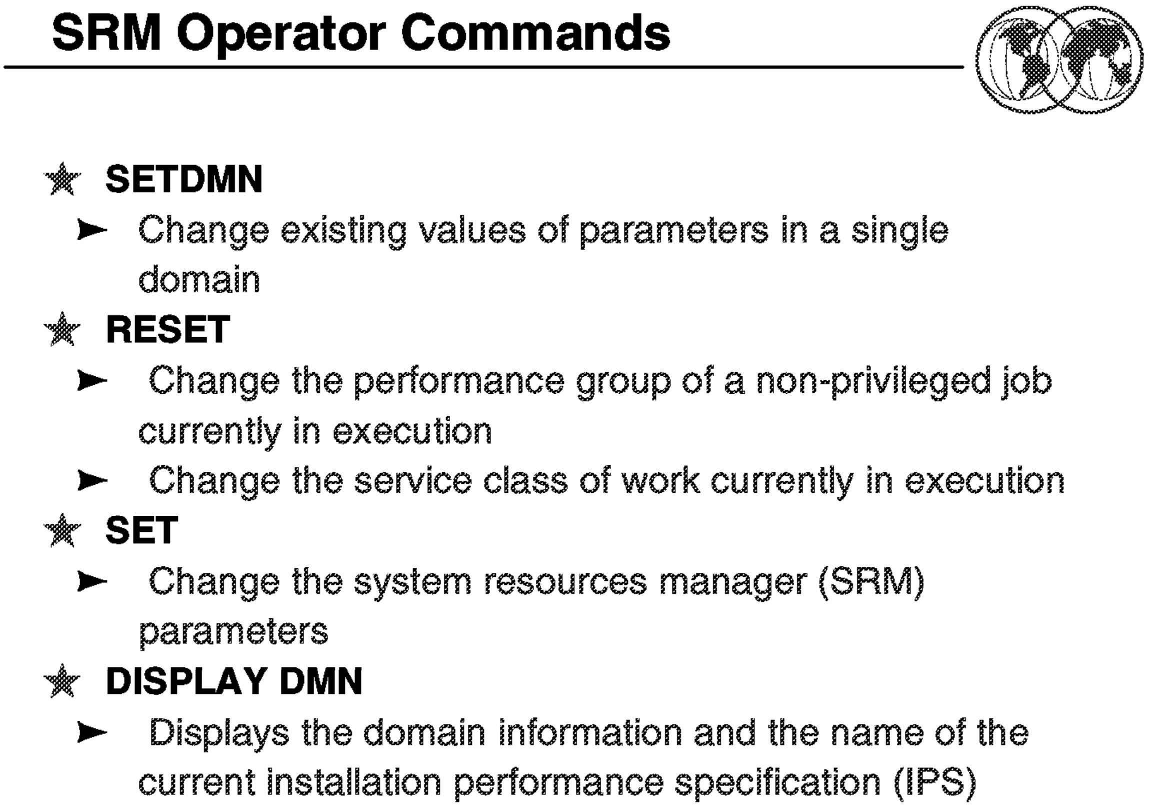
  What is the importance level of TSO/E, batch, IMS, and special batch classes in relation to one another?

  Which may be delayed or “tuned down” to satisfy other requirements?

  In other words, which response requirements are fixed and which are variable?

  What percentage of system resources should each group receive?

  Once these questions have been answered, and the installation is somewhat confident that its requirements can be met by the hardware, the installation is ready to begin writing an initial IPS, OPT, and installation control specification.



SRM Operator Commands

- **SETDMN**
  - Change existing values of parameters in a single domain
- **RESET**
  - Change the performance group of a non-privileged job currently in execution
  - Change the service class of work currently in execution
- **SET**
  - Change the system resources manager (SRM) parameters
- **DISPLAY DMN**
  - Displays the domain information and the name of the current installation performance specification (IPS)

Figure 137. Operator commands related to SRM

7.2.5 Operator commands related to SRM

The system operator can directly influence SRM′s control of specific jobs or groups of jobs by entering commands from the console:

RESET The RESET command is used to change the control performance group of an executing non-privileged job (a privileged job runs in performance group zero). The new performance group applies to all steps of the job and overrides the performance group assignment in IEAICSxx. An SMF Type 90 Subtype 30 record is written each time a RESET operator command completes successfully. The record identifies the job that was reset, the operator that initiated the command, and the change that was requested.

SETDMN The SETDMN command provides a way for altering the constraint values on a domain′s multiprogramming level. The information from this command is valid only for the life of the IPL; it does not change fields in the IPS member.

- The SETDMN command *is not valid* on systems operating in workload management *goal mode*.
- The command is supported on systems operating in workload management compatibility mode.

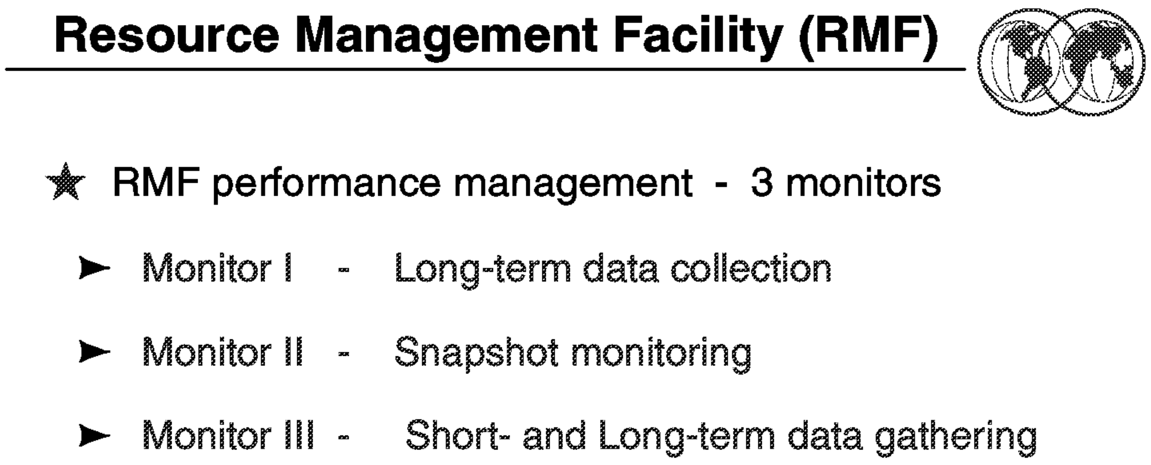
SET The SET command with the IPS, ICS, or OPT parameter is used to switch to a different IPS, installation control specification, or OPT after an IPL. SRM will base all control decisions for existing and future jobs on the parameters in the new parmlib member.

The SET IPS and SET ICS functions examine all transactions in the system and change them to the new parameters. If necessary, new transactions are started.

- The SET IPS or ICS command *is not valid* on systems operating in workload management *goal mode*.
- The command is supported on systems operating in workload management compatibility mode.

DISPLAY To understand whether the system is running effectively with the current IPS, an installation needs dynamic system status information. The DISPLAY command with the DOMAIN keyword provides a snapshot of an individual domain′s status.

- The DISPLAY DMN command *is not valid* on systems operating in workload management *goal mode*.
- The command is supported on systems operating in workload management compatibility mode.



Resource Management Facility (RMF)

- RMF performance management - 3 monitors
  - Monitor I - Long-term data collection
  - Monitor II - Snapshot monitoring
  - Monitor III - Short- and Long-term data gathering

Figure 138. Resource Management Facility (RMF)

7.3 Resource Measurement Facility (RMF)

The Resource Measurement Facility (RMF) is the strategic IBM product for performance management in an OS/390 host environment. Many different activities are required to keep your system running smoothly, and to provide the best service on the basis of the available resources and workload requirements. The operator, the administrator, the system programmer, or the performance analyst will do these tasks. RMF is the tool that helps each of these people do the job effectively. RMF gathers data using three monitors:

Monitor III Short-term data collection with Monitor III

The Monitor III gatherer session has a typical gathering cycle of one second, and consolidated records are written for a range which is typically set to 100 seconds. You can collect short-term data and continuously monitor the system to solve performance problems. You get actual performance data (response times, execution velocity) on a very detailed level for later comparison with performance policy goals. There is also information about delays, which are important indicators of performance problems. This simplifies comparison of reports created from Monitor I and Monitor III data.

Monitor II Snapshot monitoring with Monitor II

The scope of Monitor II data gathering is mainly related to single address spaces or resources, giving snapshots of the current status. You can collect data about address space activities and resource consumption, and about processor, DASD volume, and storage activities and utilization. With Monitor II, it is also possible to monitor one specific job or volume continuously.

Monitor I and III Long-term data gathering with Monitor I and Monitor III

Monitor I and Monitor III provide long-term data collection about system workload and resource utilization, and cover all hardware and software components of your system: processor, I/O device and storage activities and utilization, as well as resource consumption, activity, and performance of groups of address spaces. Data is gathered for a specific cycle time, and consolidated data records are written at a specific interval time. The default value for data gathering is one second and for data recording it is 30 minutes. You can select these options according to your requirements and change them whenever the need arises. The SMF synchronization function ensures that records are written from all monitors in the sysplex for the same intervals.

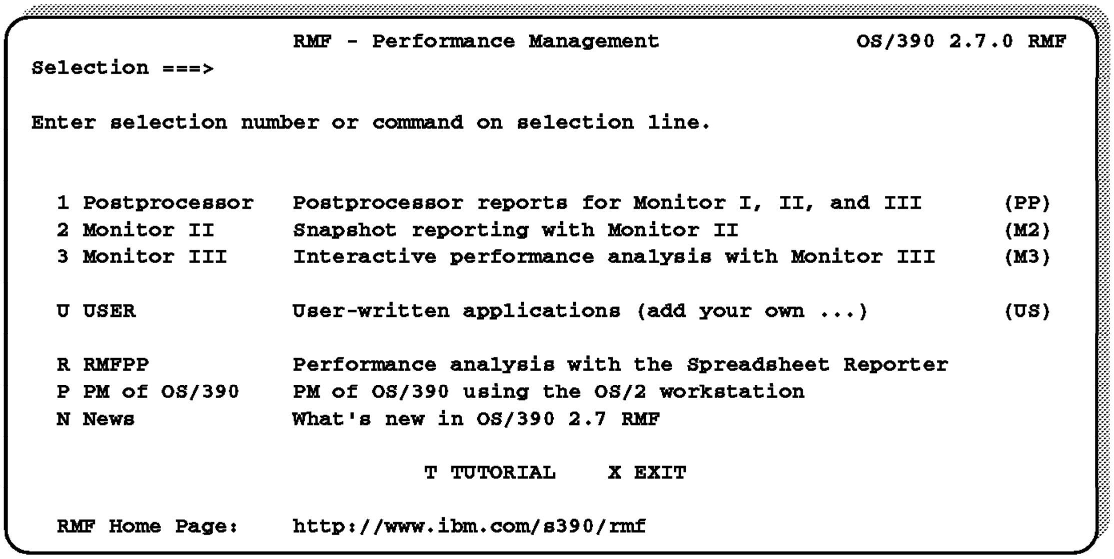
All three monitors write SMF records (type 70 -- type 79) if you define the appropriate SMF recording options. In addition, Monitor III writes VSAM records to in-storage buffers or into RMF-owned VSAM data sets.

The system operator starts all monitors as non-interactive (background) sessions with a variety of options that determine what type of data is collected and where it is stored. The data gathering functions run independently on each system, but each monitor can be started for all systems in a sysplex by one operator command. The scope of Monitor II data gathering is mainly related to single address spaces or resources, giving snapshots of the current status. With Monitor II, it is also possible to monitor one specific job or volume continuously. RMF issues reports about performance problems as they occur, so that your installation can take action before the problems become critical, such as the following:

- Determine that your system is running smoothly
- Detect system bottlenecks caused by contention for resources
- Evaluate the service your installation provides to different groups of users
- Identify the workload delayed and the reason for the delay
- Monitor system failures, system stalls, and failures of selected applications

Using RMF Monitor III

- From an ISPF command line
  - Enter the command RMF



```
                    RMF - Performance Management              OS/390 2.7.0 RMF
Selection ===>

Enter selection number or command on selection line.


  1 Postprocessor   Postprocessor reports for Monitor I, II, and III      (PP)
  2 Monitor II      Snapshot reporting with Monitor II                    (M2)
  3 Monitor III     Interactive performance analysis with Monitor III     (M3)

  U USER            User-written applications (add your own ...)          (US)

  R RMFPP           Performance analysis with the Spreadsheet Reporter
  P PM of OS/390    PM of OS/390 using the OS/2 workstation
  N News            What's new in OS/390 2.7 RMF

                              T TUTORIAL    X EXIT

  RMF Home Page:    http://www.ibm.com/s390/rmf
```

Figure 139. Using RMF Monitor III

7.3.1 Using RMF Monitor III

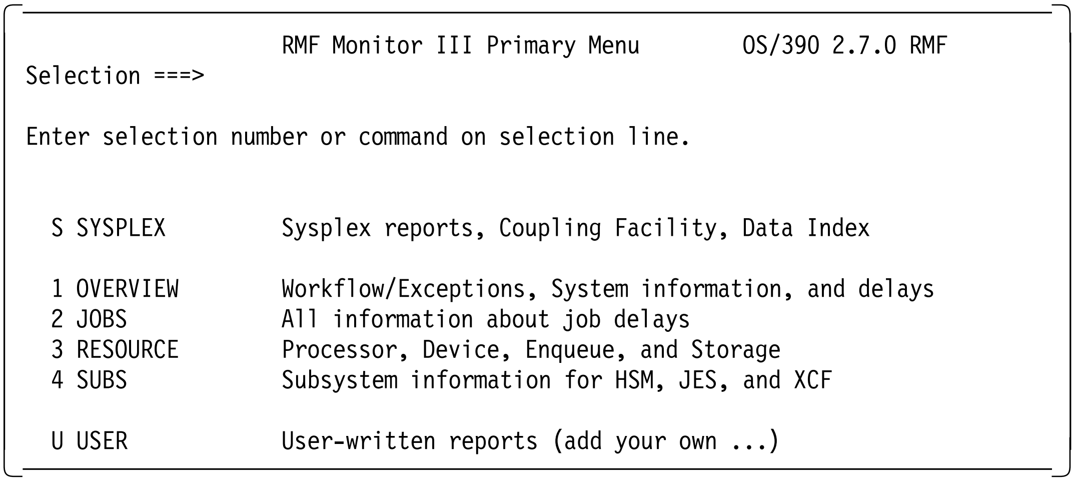
The Monitor III reporter runs in a TSO/E session under ISPF and provides sysplex or system performance reports by:

- Displaying your current system status in real-time mode
- Showing previously collected data that is still available in either in-storage buffers or preallocated VSAM data sets

To invoke RMF Monitor III, issue the command `TSO RMF` from the ISPF command line.

The visual shows the RMF Performance Management menu.

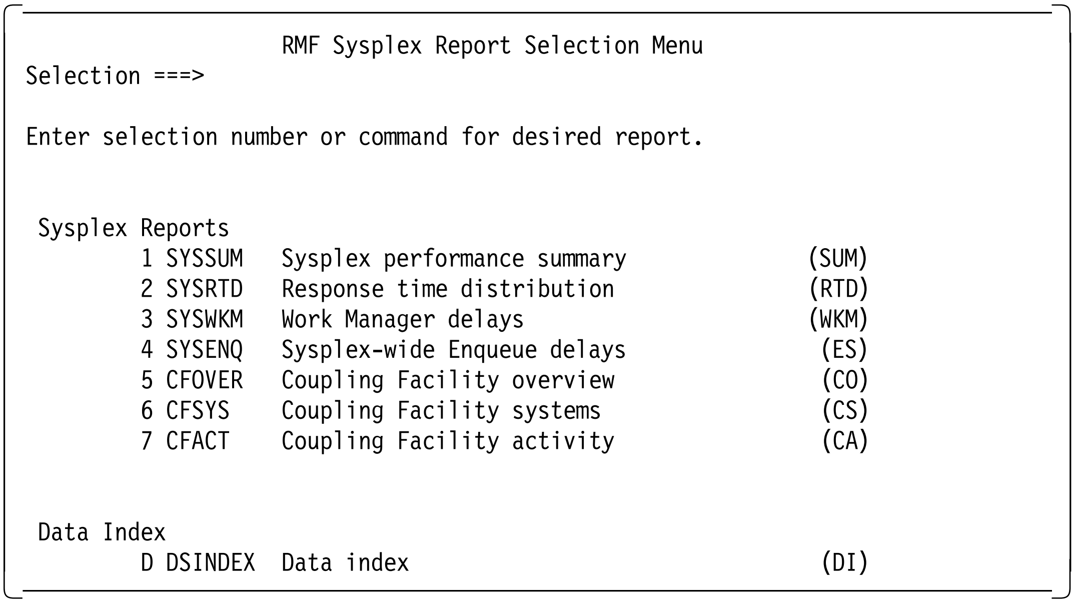
Selecting **Option 3 Monitor III** displays the RMF Performance Management Menu as shown in Figure 140 on page 227.



```
                    RMF Monitor III Primary Menu       OS/390 2.7.0 RMF
Selection ===>

Enter selection number or command on selection line.


  S SYSPLEX         Sysplex reports, Coupling Facility, Data Index

  1 OVERVIEW        Workflow/Exceptions, System information, and delays
  2 JOBS            All information about job delays
  3 RESOURCE        Processor, Device, Enqueue, and Storage
  4 SUBS            Subsystem information for HSM, JES, and XCF

  U USER            User-written reports (add your own ...)
```

Figure 140. RMF Performance Management Menu

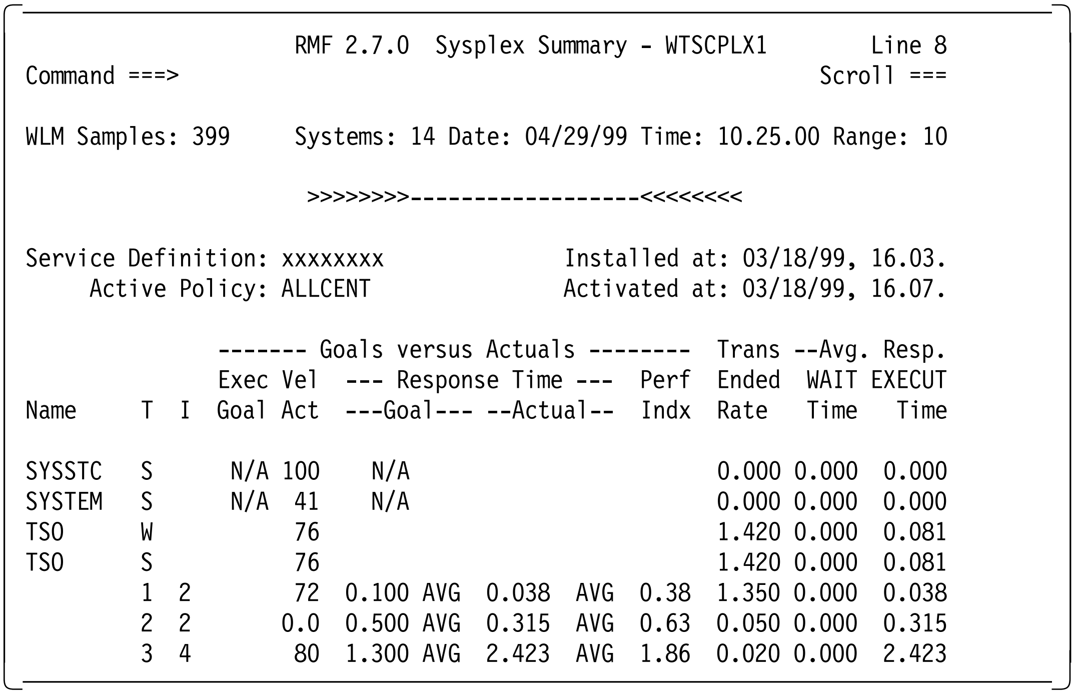
Selecting option **S SYSPLEX** displays the RMF Performance Management Menu as shown in Figure 141.



```
                    RMF Sysplex Report Selection Menu
Selection ===>

Enter selection number or command for desired report.


 Sysplex Reports
        1 SYSSUM   Sysplex performance summary            (SUM)
        2 SYSRTD   Response time distribution             (RTD)
        3 SYSWKM   Work Manager delays                    (WKM)
        4 SYSENQ   Sysplex-wide Enqueue delays             (ES)
        5 CFOVER   Coupling Facility overview              (CO)
        6 CFSYS    Coupling Facility systems               (CS)
        7 CFACT    Coupling Facility activity              (CA)


 Data Index
        D DSINDEX  Data index                              (DI)
```

Figure 141. RMF Sysplex Report Selection Menu

Figure 142 on page 228 shows a sample SYSPLEX Summary Report.

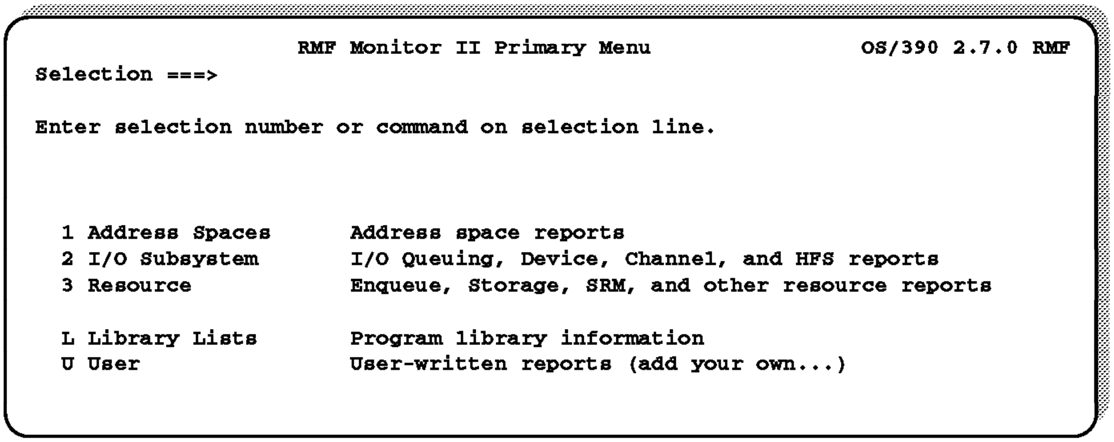


```
                     RMF 2.7.0  Sysplex Summary - WTSCPLX1        Line 8
Command ===>                                                  Scroll ===

WLM Samples: 399     Systems: 14 Date: 04/29/99 Time: 10.25.00 Range: 10

                      >>>>>>>>------------------<<<<<<<<

Service Definition: xxxxxxxx              Installed at: 03/18/99, 16.03.
     Active Policy: ALLCENT               Activated at: 03/18/99, 16.07.

               ------- Goals versus Actuals --------  Trans --Avg. Resp.
               Exec Vel  --- Response Time ---  Perf  Ended  WAIT EXECUT
Name     T  I  Goal Act  ---Goal--- --Actual--  Indx  Rate   Time   Time

SYSSTC   S      N/A 100    N/A                        0.000 0.000  0.000
SYSTEM   S      N/A  41    N/A                        0.000 0.000  0.000
TSO      W           76                               1.420 0.000  0.081
TSO      S           76                               1.420 0.000  0.081
         1  2        72  0.100 AVG  0.038  AVG  0.38  1.350 0.000  0.038
         2  2       0.0  0.500 AVG  0.315  AVG  0.63  0.050 0.000  0.315
         3  4        80  1.300 AVG  2.423  AVG  1.86  0.020 0.000  2.423
```

Figure 142. Sysplex Summary

Using RMF Monitor II

- Select Option 2 from Performance Management Menu



```
                    RMF Monitor II Primary Menu          OS/390 2.7.0 RMF
Selection ===>

Enter selection number or command on selection line.



  1 Address Spaces      Address space reports
  2 I/O Subsystem       I/O Queuing, Device, Channel, and HFS reports
  3 Resource            Enqueue, Storage, SRM, and other resource reports

  L Library Lists       Program library information
  U User                User-written reports (add your own...)
```

Figure 143. Using RMF Monitor II

7.3.2 Using RMF Monitor II

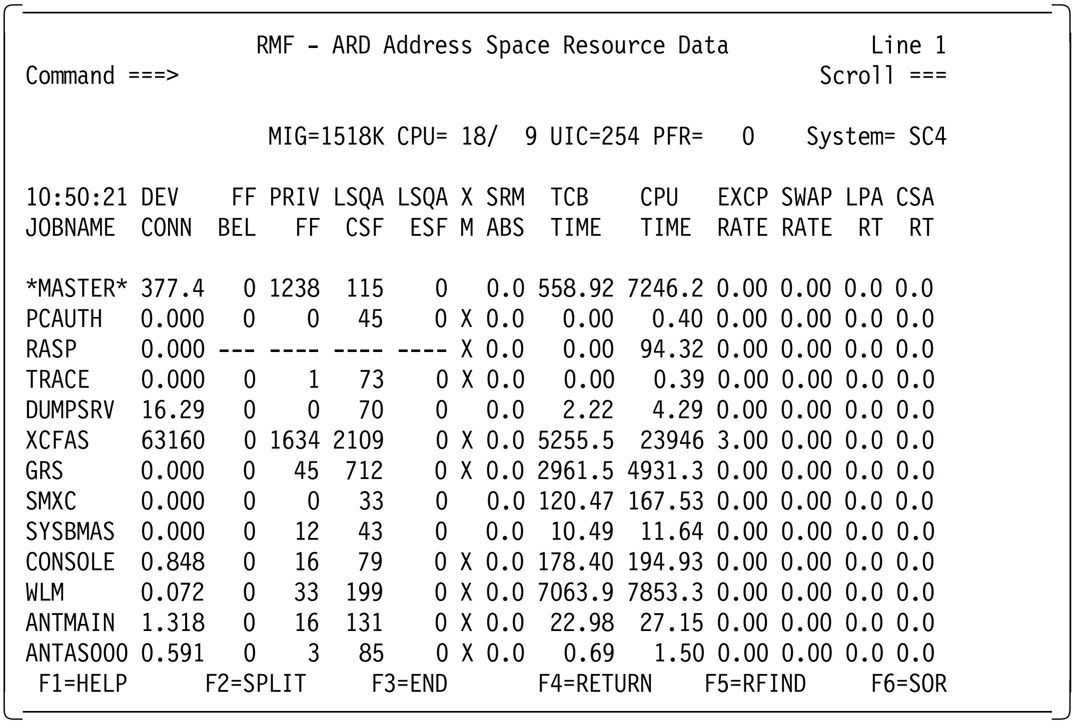
Monitor II is a snapshot reporting tool that provides very fast information about how specific address spaces or system resources (processor, DASD volumes storage) are performing. To invoke the RMF Monitor II session you can:

- Select the **Monitor II** option from the Performance Management menu.
- Issue the `TSO RMFMON` command.

Some reports offer continuous monitoring of single address spaces or DAS devices. You can get a one-line report each time you press Enter, or you can request a periodically refreshed report.

The visual displays the RMF II Primary Menu.

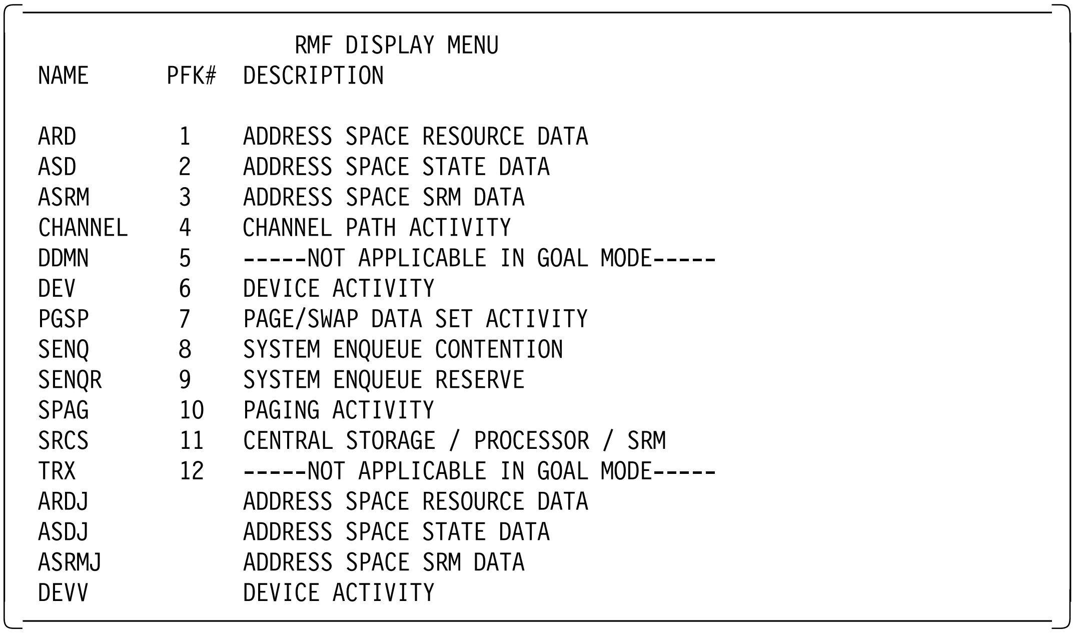
Figure 144 on page 230 displays the Address Space Resource data panel. Pressing Enter will update this display.



```
                  RMF - ARD Address Space Resource Data          Line 1
Command ===>                                                   Scroll ===

                   MIG=1518K CPU= 18/  9 UIC=254 PFR=   0    System= SC4

10:50:21 DEV    FF PRIV LSQA LSQA X SRM  TCB    CPU   EXCP SWAP LPA CSA
JOBNAME  CONN  BEL   FF  CSF  ESF M ABS  TIME   TIME  RATE RATE  RT  RT

*MASTER* 377.4   0 1238  115    0   0.0 558.92 7246.2 0.00 0.00 0.0 0.0
PCAUTH   0.000  0     0   45    0 X 0.0   0.00   0.40 0.00 0.00 0.0 0.0
RASP     0.000 --- ---- ---- ---- X 0.0   0.00  94.32 0.00 0.00 0.0 0.0
TRACE    0.000  0     1   73    0 X 0.0   0.00   0.39 0.00 0.00 0.0 0.0
DUMPSRV  16.29  0     0   70    0   0.0   2.22   4.29 0.00 0.00 0.0 0.0
XCFAS    63160   0 1634 2109    0 X 0.0 5255.5  23946 3.00 0.00 0.0 0.0
GRS      0.000  0    45  712    0 X 0.0 2961.5 4931.3 0.00 0.00 0.0 0.0
SMXC     0.000  0     0   33    0   0.0 120.47 167.53 0.00 0.00 0.0 0.0
SYSBMAS  0.000  0    12   43    0   0.0  10.49  11.64 0.00 0.00 0.0 0.0
CONSOLE  0.848  0    16   79    0 X 0.0 178.40 194.93 0.00 0.00 0.0 0.0
WLM      0.072  0    33  199    0 X 0.0 7063.9 7853.3 0.00 0.00 0.0 0.0
ANTMAIN  1.318  0    16  131    0 X 0.0  22.98  27.15 0.00 0.00 0.0 0.0
ANTAS000 0.591  0     3   85    0 X 0.0   0.69   1.50 0.00 0.00 0.0 0.0
 F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=SOR
```

Figure 144. RMF II Address Space Resource Data

Issuing the TSO RMFMON coammnd presents you with a more primitive RMF II menu system, but the reports produced are the same, and this method is often preferred for a quick reference. Figure 145 displays the TSO RMFMON Menu. Function Keys (PF keys) can be used to invoke most of the reports. To return to the menu, issue the M command. To exit RMFMON, issue the Z command.



```
                    RMF DISPLAY MENU
 NAME      PFK#  DESCRIPTION

 ARD        1    ADDRESS SPACE RESOURCE DATA
 ASD        2    ADDRESS SPACE STATE DATA
 ASRM       3    ADDRESS SPACE SRM DATA
 CHANNEL    4    CHANNEL PATH ACTIVITY
 DDMN       5    -----NOT APPLICABLE IN GOAL MODE-----
 DEV        6    DEVICE ACTIVITY
 PGSP       7    PAGE/SWAP DATA SET ACTIVITY
 SENQ       8    SYSTEM ENQUEUE CONTENTION
 SENQR      9    SYSTEM ENQUEUE RESERVE
 SPAG       10   PAGING ACTIVITY
 SRCS       11   CENTRAL STORAGE / PROCESSOR / SRM
 TRX        12   -----NOT APPLICABLE IN GOAL MODE-----
 ARDJ            ADDRESS SPACE RESOURCE DATA
 ASDJ            ADDRESS SPACE STATE DATA
 ASRMJ           ADDRESS SPACE SRM DATA
 DEVV            DEVICE ACTIVITY
```

Figure 145. TSO RMFMON Menu

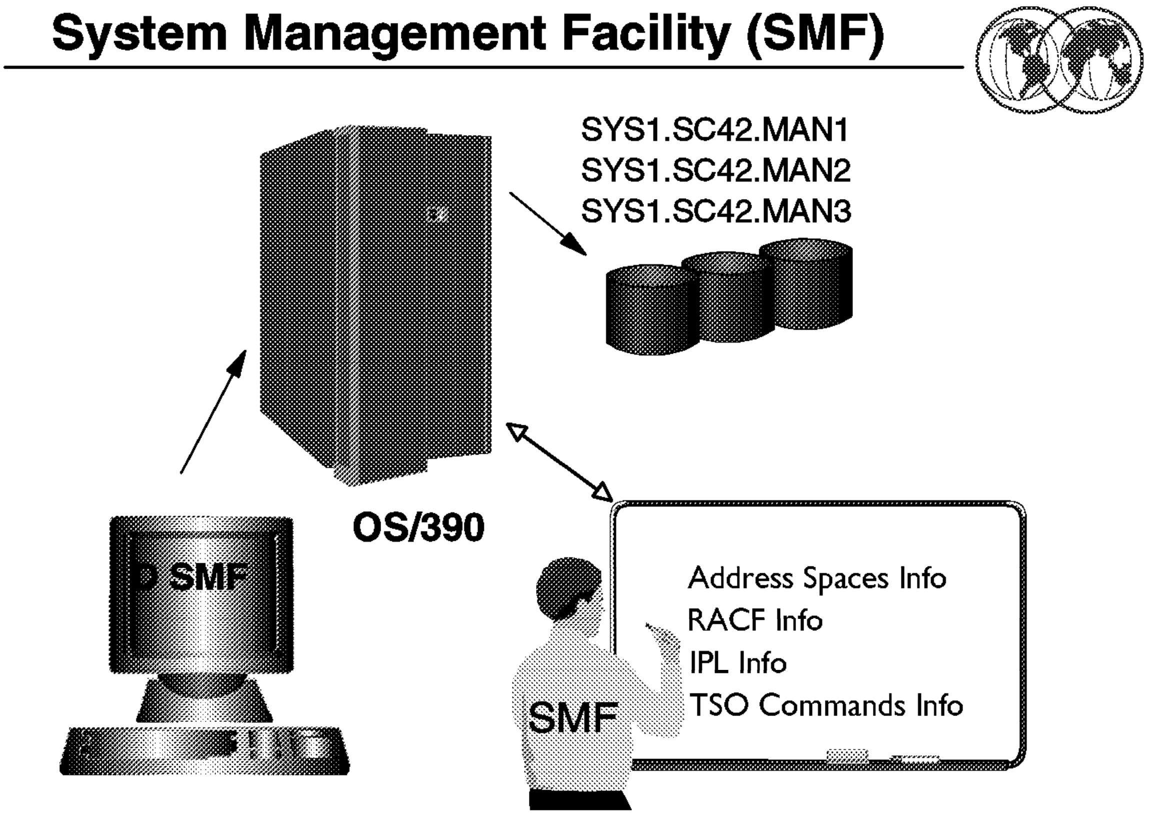


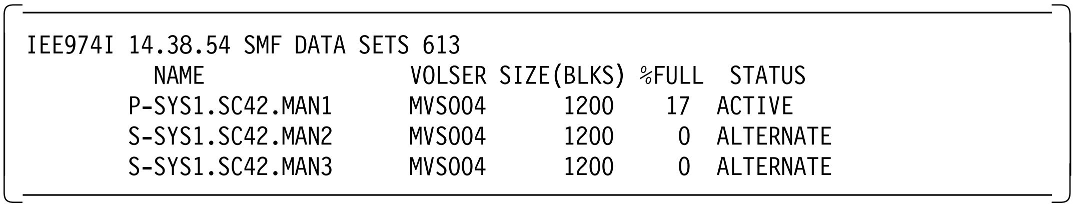
Figure 146. System Management Facility (SMF)

7.4 System Management Facility (SMF)

The OS/390 system collects statistical data for each task when certain events occur in the life of the task. SMF formats the information that it gathers into system-related records (or job-related records). System-related SMF records include information about the configuration, paging activity, and workload. Job-related records include information on the CPU time, SYSOUT activity, and data set activity of each job step, job, APPC/MVS transaction program, and TSO/E session. SMF data is written to the SYS1.MANx data sets.

SMF formats the information that it gathers into system-related records (or job-related records). System-related SMF records include information about the configuration, paging activity, and workload.

By issuing the `D SMF` command you will probably see a response similar to that in Figure 147 on page 232. This identifies three SMF data sets, one of which is ACTIVE, and it is this active SMF data set, SYS1.SC42.MAN1, that is currently storing the system and task-related performance data. When this data set fills up, the system will automatically start writing to the first available ALTERNATIVE SMF data set. The data that is stored in SYS1.SC42.MAN1 will be dumped to a data set, either on tape or DASD, using the IFASMFD utility, which will also clear this data set and make it available for reuse.



```
IEE974I 14.38.54 SMF DATA SETS 613
          NAME               VOLSER SIZE(BLKS) %FULL  STATUS
        P-SYS1.SC42.MAN1      MVS004      1200    17  ACTIVE
        S-SYS1.SC42.MAN2      MVS004      1200     0  ALTERNATE
        S-SYS1.SC42.MAN3      MVS004      1200     0  ALTERNATE
```

Figure 147. Display of SMF data sets

Customizing SMF

- ★ Use SMFPRMxx parmlib member
  - ➤ Select which SMF records are to be recorded
    - – Records have subtypes
  - ➤ Specify data set names
  - ➤ Other options
- ★ Be aware that the data sets will fill up

Figure 148. Customizing SMF

7.4.1 Customizing SMF

An installation has several ways to customize SMF to meet its needs:

- SMF parameters on the SMFPRMxx parmlib member
- Installation-written exit routines
- Operator commands

The SMF parameters allow you to:

- Record status changes
- Pass data to a subsystem
- Select specific records
- Specify data set names
- Specify the system identifier to be used in all SMF records
- Select specific subtypes
- Perform interval accounting
- Collect SMF statics
- Perform TSO/E command accounting
- Perform started task accounting

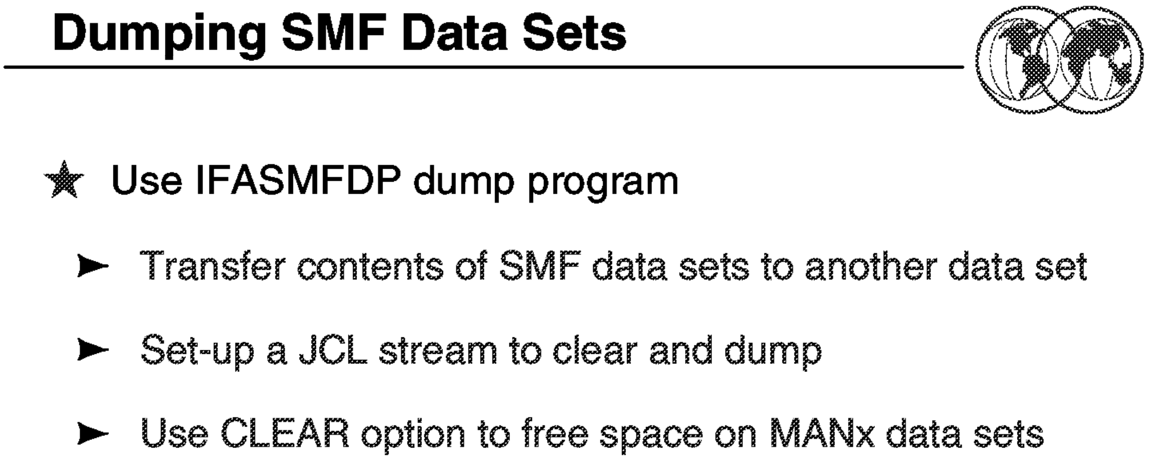
SMF records are selected by specifying either the type desired (or the types not desired) with the TYPE or NOTYPE option of the SYS or SUBSYS parmlib parameter.

If any one of record types 14, 15, 17, 62, 63, 64, 67, or 68 is specified with the TYPE option, data is collected for all records. Likewise, if either record type 19 or 69 is specified with the TYPE option, data is collected for both records. However, only those records that are selected by TYPE or NOTYPE request are written to the SMF data set.

Installation-written exit routines IEFU83, IEFU84, and IEFU85 (SMF writer) and IEFACTRT (termination) can control which records are to be written to the SMF data set. After inspecting an SMF record, these routines return a code to the system indicating whether the record is to be written to the SMF data set.

The TYPE option of the SYS and SUBSYS parameter provides inner parentheses to indicate the subtypes to be collected. If subtype selection is not used, the default is all subtypes. The following is an example of how the TYPE option should be used to record subtypes 1, 2, 3, 5, 6, 7, and 8 for the type 30.

```
    SYS(TYPE(30(1:3,5:8)))  or  SUBSYS(STC,TYPE(30(1:3,5:8)))
```



Dumping SMF Data Sets

- Use IFASMFDP dump program
  - Transfer contents of SMF data sets to another data set
  - Set-up a JCL stream to clear and dump
  - Use CLEAR option to free space on MANx data sets

Figure 149. Dumping SMF data sets

7.4.2 Dumping SMF data sets

SMF data sets cannot be shared across systems. IBM does not recommend that the SMF dump program be run from one system in an attempt to clear SMF data set used by another system.

When notified by the system that a full data set needs to be dumped, you use the SMF dump program (IFASMFDP) to transfer the contents of the full SMF data set to another data set, and to clear the dumped data set so that SMF can use it again for recording data.

The SMF dump program dumps the contents of multiple VSAM or QSAM data sets to sequential data sets on either tape or direct access devices. The SMF dump program allows the installation to route different records to separate files and produce a summary activity report. Figure 150 shows a sample job stream to dump and clear the SMF data set.

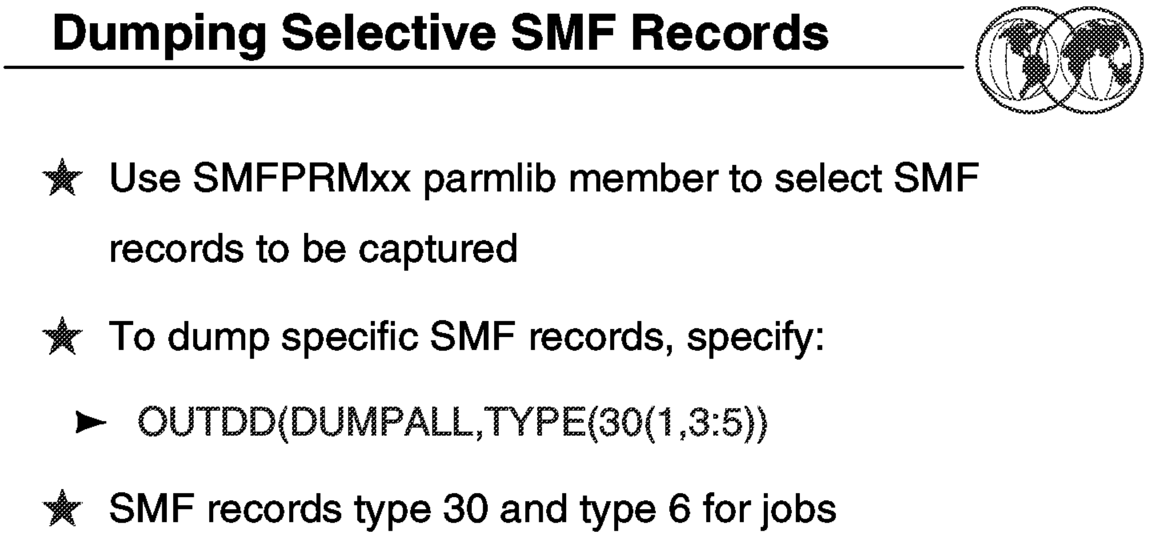
```
//STEP1   EXEC PGM=IFASMFDP,REGION=4M
//DUMPIN   DD  DSN=SYS1.SC42.MAN1,DISP=SHR
//DUMPALL  DD  DISP=(,CATLG),DSN=SMFDATA.BACKUP(+1),
//             UNIT=SYSALLDA,VOL=SER=SMF002,SPACE=(CYL,(30,10),RLSE),
//             DCB=SMFDATA.ALLRECS.GDGMODEL
//SYSIN    DD  *
INDD(DUMPIN,OPTIONS(ALL))
OUTDD(DUMPALL,TYPE(000:255))
```

Figure 150. IFASMFD job stream to dump and clear the SMF data set

In Figure 150 the **INDD(DUMPIN,OPTIONS(ALL))** parameter indicates that the SMF data set will be dumped and cleared. To process only a DUMP request, set **OPTIONS(DUMP)**. To process only a CLEAR request, set **OPTIONS(CLEAR)**.

The **OUTDD(DUMPALL,TYPE(000:255))** parameter indicates that SMF records in the range of 000 to 255 will be dumped (in other words, everything). If you only wanted to dump TYPE30 Subtype 1, 3, 4 and 5, you would code **TYPE(30(1,3:5))**.

Alternatively, if you did not want to dump these records, you would code **NOTYPE(30(1,3:5))**.



Dumping Selective SMF Records

- Use SMFPRMxx parmlib member to select SMF records to be captured
- To dump specific SMF records, specify:
  - OUTDD(DUMPALL,TYPE(30(1,3:5))
- SMF records type 30 and type 6 for jobs

Figure 151. Dumping selective SMF records

7.4.3 Dumping selective SMF records

Subtype selectivity for SMF records is an option of the TYPE or NOTYPE option on the SYS and SUBSYS parameters used for SMF recording. Subtype selectivity allows more flexibility in post-processing records and helps control the amount of data stored in SMF data sets.

The subtype selectivity function supports only those records which use the standard SMF record header. The header requires the subtype field is at offset 22 (decimal) and the ′subtypes used′ bit (bit position 1), of the system indicator byte at offset 4 (decimal), is set to X′1′. SMF processes the record for subtype selectivity when both conditions are met. This support is limited to SMF record types 0 through 127 (user records are not supported).

7.4.3.1 SMF type 30 records

For type 30 SMF records, there are five subtype records that provide address space level accounting information. SMF writes a subtype 1 record at the start of a work unit (such as a job), a subtype 2 record at the end of each recording interval, subtype 3 and 4 records at the end of a job step, and a subtype 5 record at the end of a work unit (such as a job).

A job may consist of one or more job steps. SMF type 30 records are as follows:

Subtype 1 Job Initiation - SMF Record TYPE30 Subtype 1

Job initiation begins when MVS receives the job from JES, and then MVS initiates the first step. SMF Record type 30 subtype 1 is written at job initiation, TSO/E logon, STC and APPC task initiation. This record contains the following information:

- JES job selection priority
- JES-assigned JOB/TSU/STC number
- Job initiate time stamp
- Job name (Batch) or Logonid (TSO)
- Job class
- MVS Product level that created this record
- Performance group assigned to this job at initiation
- Programmer name field from JOB statement
- RACF group identification
- RACF terminal name used which indicates the VTAM terminal name used for this TSO session
- RACF user identification
- Subsystem value may be either JES2 or JES3
- Identification of the system on which the job initiated
- Type of task
  - A APPC session
  - JOB batch job
  - TSU TSO session
  - STC started (system) task

In a JES2 environment, the first step of every job begins by issuing the data set enqueue request for every data set name in all steps of the job. JES3 issues its enqueue requests before initiation.

If the data set name is not available for this job because another task has an exclusive enqueue (which is usually requested as DISP=OLD on the DD card and means “I do not want to let anyone else reference this data set while I am using it”), or if this job has requested an exclusive enqueue, the job enters a data set enqueue delay state and the operator is notified of the conflict with a console message:

```
FOLLOWING DATA SETS RESERVED OR UNAVAILABLE.
```

If the operator takes no action, the job waits, tying up an initiator but using no other resources, until the job or TSO session using the dataset completes and releases the data set.

If the operator cancels the job, MVS initiation is terminated, and both SMF TYPE30_4 step record and TYPE30_5 job record are written (both indicate SYSTEM ABEND 222).

Once the data sets enqueue process completes, step allocation commences, during which all devices required by this step are acquired, one at a time, in the order of the DD statements for the step. Allocation is the process that connects the DDNAME that your program will use to the actual physical location of the dataset.

For an existing DASD data set, the allocation process finds the UCB address of the volume and reads the VTOC to find the cylinder, head, and record number that identifies the start of the data set. If the request is for a new DASD data set, the allocation process selects a VOLSER, as specified in the JCL, or automatically as per the SMS policies, finds free space on that volume, and creates the VTOC entry for the new data set.

If a tape drive is needed and one is available, the allocation process assigns a specific tape drive address to your job. If all online tape drives are already being used by other jobs, the job enters Allocation Recovery and the operator is notified by console message:

```
UNABLE TO ALLOCATE 1 DRIVE.  THESE DRIVES ARE OFFLINE: 181, 182
   REPLY DEVICE NAME, WAIT, OR CANCEL
```

Allocation is complete when all needed devices have been given to the job. Mounts for the first tape volume on each tape drive are issued at the end of allocation, but allocation does not wait for tape mount completion.

When allocation for this step is complete, the program to be executed is loaded into memory by Program FETCH and the date and time of the program load is stored. Although the job has theoretically been active since initiation, it is only now, after the program load, that your program can execute its instructions, use CPU time, and perform input/output functions. When this program completes its work (or ABENDs!), the termination date and time is stored, as well as the step completion code.

Subtype 4 Step Termination - SMF Record TYPE30 Subtype 4

At step termination an SMF TYPE30 Subtype 4 record is created which contains information related to the resources used for the entire step. SMF TYPE30 Subtype 4 data includes:

- Step completion indicator
- ABEND reason code
- Step active time
- Allocation time stamp
- Step allocation time
- Condition code
- CPU time used by the Second Level Interrupt Handler (SLIH) in servicing of I/O requests for this task
- DASD volumes mounted for specific volume requests
- Dispatching priority at step termination
- Duration of step between initiation and beginning of allocation
- EXCPs records for each device type
- EXCPs delivered to this task, as counted by RMF
- Initiation time stamp
- Job number assigned by JES
- Job name (batch or APPC); user ID (TSO), started task name (STC)
- Program Load time stamp
- Program name (PGM=) from EXEC statement
- Region size
- Step name appearing on the EXEC PGM= statement
- TAPE volumes mounted for specific volume requests

If the job has additional steps, each step will initiate, allocate, load and terminate individually, until all steps have run. Each step completion will create an SMF TYPE30 Subtype 4 record.

Subtype 5 Job Termination - SMF Record TYPE30 Subtype 5

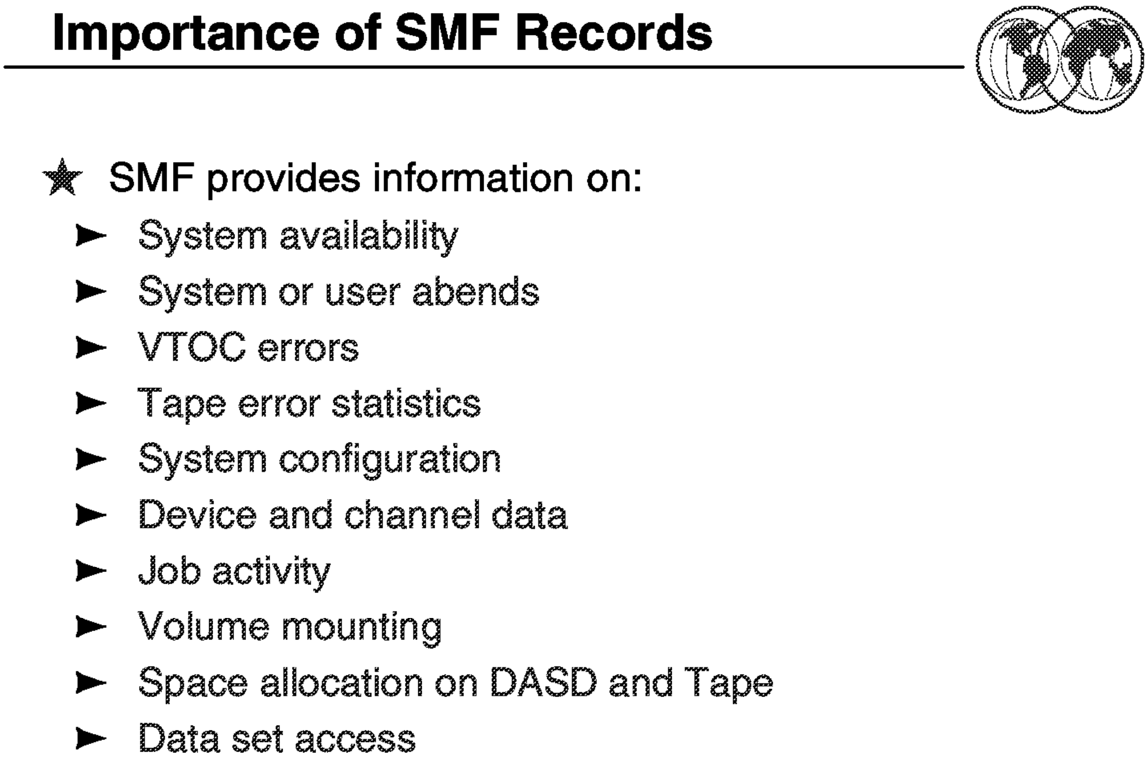
When the final step terminates, control is given back to OS/390 for job termination and an SMF TYPE30 Subtype 5 record is created. The SMF TYPE30 Subtype 5 record contains the JOB total resources, whereas the TYPE30 Subtype 4 record contains step resources. Almost all of the variables are identical in name and description to the Subtype 4 records.

7.4.3.2 SYSOUT Processing - SMF Record TYPE6

After the job has executed and created any SYSOUT (output) files, JES will process the output. JES2, JES3, PSF, External Writers, and some vendor products create TYPE6 SMF records when a SYSOUT DD′s spool records are processed. A separate TYPE6 record is created for each Job Output Element (JOE). The TYPE6 record is written at the completion of each printing event, which normally occurs after job completion. The TYPE6 record contains the number of logical records actually sent to the printer device. For non-APF output, this will be the number of actual number of print lines sent to the printer device. Some of the SMF TYPE6 variables include:

- Form number
- Job number assigned by JES
- Number of logical records actually physically processed by JES
- SYSOUT class
- Output device name
- Approximate page count
- Print or punch started time stamp
- Print or punch ended time stamp

The SMF data that is collected for a single job enables you to analyze all aspects of that job and how the system processes the job′s requirements. Multiply the amount of information that is collected for one job, by all the tasks running in the system, and you can see that we are building an extensive source of data that can be used to help us understand the way the system is functioning, and more importantly assist us with making decisions about improving the performance of the system.



Importance of SMF Records

★ SMF provides information on:

- System availability
- System or user abends
- VTOC errors
- Tape error statistics
- System configuration
- Device and channel data
- Job activity
- Volume mounting
- Space allocation on DASD and Tape
- Data set access

Figure 152. Importance of SMF records

7.4.4 Importance of SMF records

We have briefly reviewed the job-related TYPE30 SMF records, but there are different SMF records created for all functions within the OS/390 system and its related subsystems. The information that is contained within each SMF record is extensive, and covers most aspects of task and system functions. We will briefly discuss some of the information that can be obtained by reviewing the SMF records, which when reviewed over a larger time frame, build an historical database that can be used to plot trends and enable you to prepare for future workload and threshold problems.

- System Availability - SMF produces records at IPL time and when the operator enters a HALT EOD command preceding the scheduled shutdown of the system. By examining these records and the last SMF record recorded before shutdown of the system, an installation can establish the following information for a given time period.

  - Reporting interval number of IPLs
  - System up time and system down time.
  - Number of scheduled stoppages and the approximate amount of scheduled down time
  - Number of unscheduled stoppages and the approximate amount of unscheduled down time
  - Reasons for system failure

  In addition, JES2 and JES3 produce the SMF subsystem start (type 43) and subsystem stop (type 45) records. From these records, an installation can further analyze the system′s availability by checking the start time, stop times and circumstances under which JES2 or JES3 was started (for cold start versus a warm start).

- Abend Code Summary - SMF reports a system or user abend (abnormal end task) code for each job (and job step) that abends. By tracking those codes issued by operational procedures (such as codes 122 and 222 for operator cancels), an installation can account for any loss of CPU time due to job reruns. More generally, a summary of the abend codes by program name or code allows an installation to determine which programs are abending frequently and which codes are occurring most often. This information might show the need for software error corrections, JCL revisions, or better operating instructions.

- Direct Access VTOC Errors - The SMF record type 19 has a VTOC indicator bit that the system sets if there is a failure while updating a VTOC (volume table of contents). By checking the setting of this bit, operations personnel can identify any VTOCs that might have missing tracks or overlapping data sets.

- Tape Error Statistics SMF record type 21 provides tape error statistics such as the number of temporary read and write errors, permanent read and write errors, noise blocks, erase gaps, and cleaner actions. By sorting and summarizing these error statistics by tape volume (or tape unit), operations personnel can identify volumes that might need reconditioning or replacement, or point out tape drives that might require cleaning or maintenance.

- Analyzing the Configuration - SMF generates records that describe changes in the system configuration:

  - At IPL for online devices (types 0, 8, 19, and 22)
  - When a device is added to the configuration (types 9 and 10)
  - When a device is removed from the configuration (type 11)
  - When a processor, channel path, storage device moves online or offline (type 22)

  In addition to these records, operations management can use specific information in other SMF records to report configuration statistics. The following examples show this use of SMF:

- Device and Channel Loading - From SMF records, an installation can obtain the total problem program EXCP counts by device and by channel over a given reporting period.

- Concurrent Device Use - An installation can combine the data in the SMF step termination records to report the number of devices per device type that problem programs used during specified intervals. By using this report with the RMF device activity records (type 74), an installation can identify periods of the day when the percentage of problem program device use was exceptionally high or low.

- Concurrent Job Activity - The SMF job initiation and termination record contain the start and stop times of each batch job, job step, TSO/E session, APPC/MVS transaction program, and started task. Using these times, an installation can determine the jobs that are running during the same interval. Evaluation of this data might assist with optimizing system performance by rescheduling some of the workload.

- Job Wait Time in Initiation and SYSIN/SYSOUT Queues - The SMF step termination records have the following three time stamps: step initiation time, device allocation start time, and problem program start time. By calculating the differences in these three times, an installation can identify any abnormally long job step initiation.

- Allocated But Unused Direct Access Storage - There are many times when users make allocation requests for direct access storage that exceed the actual requirement. This misuse can be a significant drain on the direct access resource pool.

- Volume Mounting - SMF writes a type 19 record whenever a volume that is defined by a DD statement is demounted. Summarizing these records by volume can give an installation some indication of its direct access volume mounting activity for problem programs.

- Fragmented Volumes - Periodic analysis of the SMF type 19 records can be useful in identifying direct access volumes whose unallocated space is fragmented.

- Evaluating Data Set Activity SMF produces several records that contain information on data set activity. These records, which include types 4, 14, 15, 17, 18, 30, and 34, can help the installation to answer the following questions:
  - What is the average data set size for both tape and direct access devices?
  - Is the number of multi-volume data sets significantly large?
  - What percentage of all data sets are permanent? What percentage are temporary?
  - What percentage of all temporary data sets does VIO (virtual input output) control?

  An installation can identify the volumes that might need reorganization by examining the relationship of the following SMF fields

  - The number of unallocated cylinders and tracks
  - The number of cylinders and tracks in the largest unallocated extent
  - The number of unallocated extents
- Multiple Extents - By checking the number of extents field in the UCB section of SMF type 14 and 15 records, an installation can identify direct access data sets that have exceeded their primary allocation and have used secondary allocation(s). Although useful, secondary allocation might affect system performance and fragment the space on direct access volumes.
- Data Set Modifications - SMF generates a record each time a user:
  - Scratches a non-VSAM data set (type 17)
  - Renames a non-VSAM data set (type 18)
  - Updates a VSAM data set (type 60)
  - Defines a catalog entry for the integrated catalog facility (type 61)
  - Alters or renames a catalog entry for the integrated catalog facility (type 66)
  - Defines or alters a VSAM catalog entry (type 63)
  - Deletes a catalog entry for the integrated catalog facility (type 65)
  - Deletes a VSAM catalog entry (type 67)
  - Renames a VSAM catalog entry (type 68)
- Open/Close Activity - SMF writes a type 14 or 15 record whenever a data set is closed or processed by EOV. The installation can determine how many times EOV closed or processed a given data set by counting the number of type 14 and 15 records.
- Blocking Factors - By examining the block size and logical record length fields recorded in the SMF type 14 and 15 records, an installation can identify those data sets that the system is processing with ineffective blocking factors.

Most SMF records contain general identification fields such as the job name, step name, programmer name, reader start time, and reader start date. By sorting and summarizing SMF data according to these types of fields, an installation can create reports or profiles that show each batch job, job step, and TSO/E session′s use of system resources such as:

- CPU time
- Storage
- Paging facilities
- I/O devices
- Service units
- Programming languages

Chapter 8. OS/390 problem diagnosis

MVS can process large amounts of work efficiently because it keeps track of storage in a way that makes its storage capacity seem greater than it is. It′s a complex system made up of many components, similar to the human body. And, like the human body, MVS can experience problems that need to be diagnosed and corrected.

If an MVS problem occurs, this redbook can help you determine what happened and why it happened. The following are examples of problems you might encounter while running MVS:

- An abnormal end occurs in processing, known as an abend.
- A job remains hung in the system.
- The system or a process repetitively loops through a series of instructions.
- Output looks nothing like you thought it would.
- Processing slows down.
- Processing stops, requiring that you reIPL.

For system problems, MVS displays symptoms that will help you with your diagnosis. Problem source identification, called PSI, is the determination of what caused the error. Why was an abend issued? What program is using so much of the system storage? What component caused that hang? Which program is looping?

MVS supplies many tools and service aids that assist with problem diagnosis. These tools includes dumps and traces, while service aids includes the other facilities provided for diagnosis. For example:

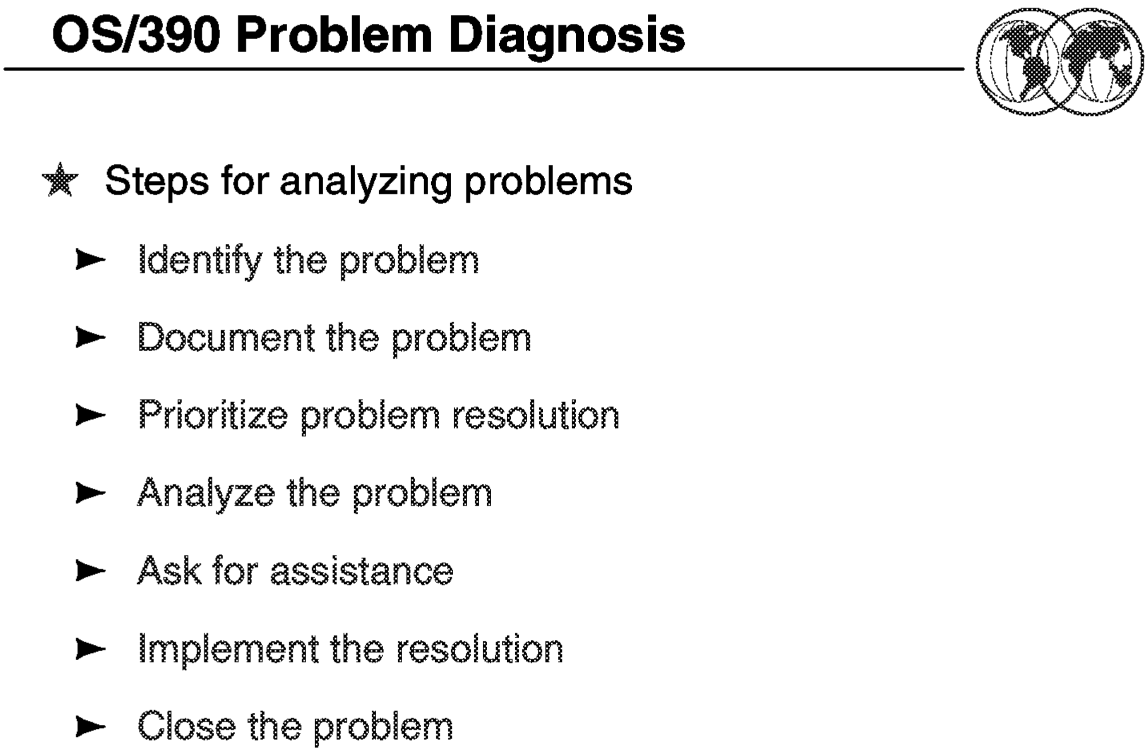
- SVC dump and system trace are tools
- Logrec data set and AMBLIST are service aids

The interactive problem control system (IPCS) is a tool provided in the MVS system to aid in diagnosing software failures. IPCS provides formatting and analysis support for dumps and traces produced by MVS, other program products, and applications that run on MVS.

This chapter is for anyone who performs diagnosis of software on an MVS system. Usually, this person is a systems programmer. When using IPCS as an aid in dump and trace analysis, the programmer should be able to:

- Understand basic MVS system concepts
- Code JCL statements to run programs or cataloged procedures
- Code in assembler language and read assembler, loader, and linkage-editor output
- Understand commonly-used diagnostic tasks

This chapter describes some basic fundamentals to diagnose the system problems. There are times, however, when your diagnosis will require the assistance of the IBM Support Center.



OS/390 Problem Diagnosis

- Steps for analyzing problems
  - Identify the problem
  - Document the problem
  - Prioritize problem resolution
  - Analyze the problem
  - Ask for assistance
  - Implement the resolution
  - Close the problem

Figure 153. OS/390 problem diagnosis fundamentals

8.1 OS/390 problem diagnosis fundamentals

This chapter will discuss problem diagnosis fundamentals and will include analysis methodologies for the OS/390 system, as well as techniques that will assist with the analysis of Language Environment, CICS, CICSPlex/SM, MQSeries, VTAM, and DB2 problems.

At the completion of this chapter you will be able to:

- Adopt a systematic and thorough approach to dealing with problems
- Identify the different types of problems
- Understand where to look for diagnostic information and how to obtain it
- Interpret and analyze the diagnostic data collected
- Escalate the problem to the IBM Support Center

The steps that are taken to investigate and analyze a problem are shown on the visual.

8.1.1 Identifying the problem

A system problem can be described as any problem on your system that causes work to be stopped or degraded. The steps involved in diagnosing these problems are different for each type of problem. Before you can begin to diagnose a system problem, however, you have to know what kind of problem you have. Problem identification is often not a straight-forward process, but an investigative exercise that requires a structured method that will enable the correct initial assessment to be made. This initial phase is important because decisions you make now relating to diagnostic data collection will influence the speed of the resolution.

The most important questions you must ask, include:

- Is the process that is causing the problem a new procedure, or has it worked successfully before?
- If it was an existing procedure that was previously successful, what has changed?
- What messages are being generated that could indicate what the problem is? These could be presented on the terminal if the process is conversational, or in the batch or subsystem joblog, or in the system log (SYSLOG).

  Note: Review the *OS/390 MVS System Messages* and *OS/390 MVS Systems Codes*, GC28-1780.
- Can the failure be reproduced, and if so what steps are being performed?
- Has the failing process generated a dump?

All of these questions will enable you to develop an appropriate plan of action to aid with the resolution. You can never be criticized for providing too much diagnostic data, but too little information only delays the solving or escalation of the problem.

8.1.2 Document the problem

Documentation of the problem and the analysis steps taken can assist with not only initial resolution, but will also assist if the problem occurs again. For larger more complex problems regular documentation during the analysis process can highlight areas that will become more crucial as the investigation progresses. This will enable you to develop a flow chart and reference point listing that can be referred to throughout your analysis. Document the final resolution for future reference.

8.1.3 Prioritize problem resolution

Your prime objective as a system programmer is to ensure system availability, and in the event of a major subsystem failure, for example, a Customer Information Control System (CICS) failure, or worse still the whole OS/390 system, your focus will be on the speedy restoration of the system.

Subsystem failures will often generate their own diagnostic data, and the recovery process is often fairly straight forward. These systems will generally perform cleanup processes during recovery and system availability will be resumed. If the subsystem fails during the recovery then immediate problem analysis and resolution will be required.

The worst-case scenario is that your complete OS/390 system is down. Swift system recovery is required, but a decision must be made to determine whether the currently preserved central storage should be dumped via a stand-alone dump routine prior to the recovery Initial Program Load (IPL). The IPL process will clear central storage, therefore any failure information will be lost. The stand-alone dump process will take some time but could be extremely valuable should the problem re-occur.

8.1.4 Analyze the problem

Before you start the process of what could be described as “the more complex analysis procedures,” you should review all of the data you currently have that may solve your problem. Have you:

1. Looked in the system log for any relevant messages or abend information.
2. Looked in the job log for any relevant messages or abend information.
3. Reviewed the meanings of any messages or codes in the relevant manuals.
4. Reviewed the system error log, SYS1.LOGREC, which contains information about hardware and software failures.

Problem analysis is, like any process, a skill that develops the more you use it. Unfortunately, problems vary in their complexity and frequency, and it is possible that tasks requiring this type of review may be infrequent in your environment. Of course the ultimate aim is to have little need to perform complex problem diagnosis. This is why a sound methodology is necessary to assist with your analysis.

It is necessary to retain a focus during the analysis process and be aware that there are often alternative ways to approach a problem. To ask for assistance with a problem is not a sign of failure, but an indication you are aware that another person′s views could speed up the resolution. A fresh idea can often stimulate and enhance your current thought processes.

Solving a problem is a combination of:

1. Your ability to understand the problem.
2. The quality of the diagnostic data you can review.
3. Your ability to use the diagnostic tools at your disposal.

8.1.5 Ask for assistance

You will hopefully be aware if you are making little progress with your diagnosis that some assistance may be required. What you, and your manager are seeking is a speedy resolution, and it is far more professional to use all the facilities and technical expertise available to assist you. The IBM Support Center is there to assist you with your problems and the diagnostic data you have collected, and the analysis steps you have already performed will be of great help to the Support Center when they review the problem.

8.1.6 Implement the resolution

Successful diagnosis of the problem will result in a number of possible resolutions:

- *User Error* - This will require the user to correct their procedure to ensure a satisfactory resolution is implemented. If their procedure is impacting other users, then it is imperative that their prompt action be encouraged.
- *Software implementation error* - You must ensure that all installation procedures have been correctly executed and any required customization has been performed correctly. Until you can be sure of a successful implementation it is advisable to remove this software, or regress to a previous level of the software until more extensive testing can be done in an environment that will not impact production workloads.
- *Software product fault* - If the fault is identified as a failure in software a fix might already have been developed to solve this problem. This fix is identified as a Program Temporary Fix (PTF) and will need to be installed into your system. If the problem is causing a major impact, it is suggested that you expedite your normal migration process and promote the fix to the problem system to hopefully stabilize that environment.

If the problem has not been previously reported an authorized program analysis report (APAR) will be created and a PTF will be generated.

- *Hardware fault* - This is the resolution that will be controlled by the hardware service representative, but may require some reconfiguration tasks depending on the nature of the problem. Consultation with the hardware vendor′s service representative will clarify the requirements.

8.1.7 Close the problem

When you have tested and implemented the problem resolution, ensure that all parties involved with this problem are informed of the closure of this issue.

It should be noted that during your career you will experience some problems that occur only once, and even with the best diagnostic data, cannot be recreated or solved, by anyone. When this happens there is a point in time where you must accept the fact that this anomaly, was in fact, just that, an anomaly.

Program Status Word (PSW)

- GOVERNS PROGRAM CURRENTLY EXECUTING
  - CONTROLS INSTRUCTION SEQUENCING
  - DETERMINES STATE OF CP
- PROVIDES INTERRUPTION CAPABILITY

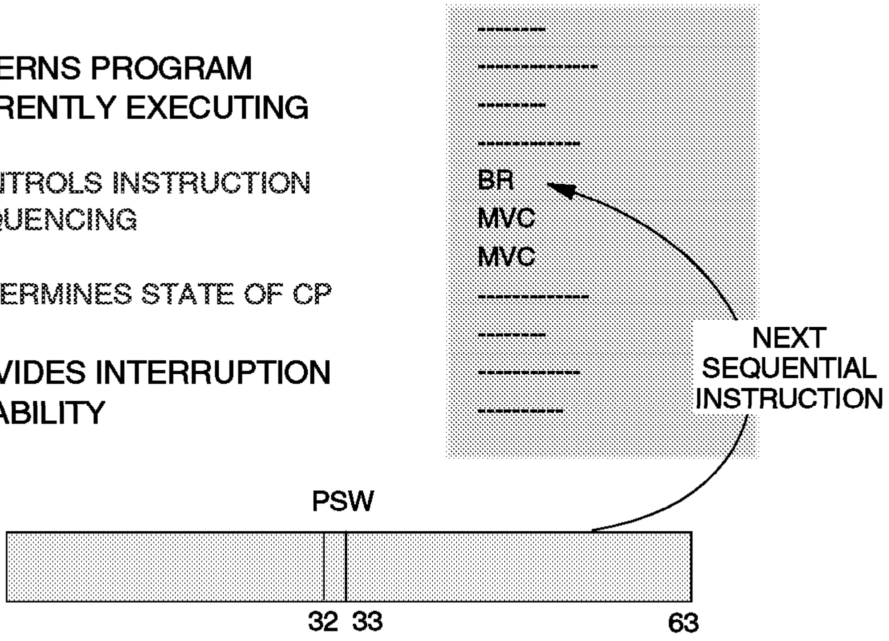


Figure 154. Program Status Word (PSW)

8.2 Program Status Word (PSW)

One very important piece of information that will be crucial to your ability to diagnose a problem is the program status word, more commonly referred to as the *PSW*. The PSW includes the instruction address, condition code, and other information to control instruction sequencing and to determine the state of the CPU. The active or controlling PSW is called the current PSW.

The PSW is so important because it keeps track of the progress of the system and the executing program. More importantly the current PSW points to the address of the next instruction to be executed. In some specific cases the PSW will point to the failing instruction, but in most cases it will point to the next instruction.

What this means is that when a task abends and a dump is taken, the PSW is pointing to the next instruction that will be executed in the failing program. By subtracting the instruction-length code (ILC) from the PSW address, we will most likely be looking at the failing instruction that caused the abend.

Note: For page translation and segment translation errors the PSW can actually point to the failing instruction.

8.2.1 What is addressability

One of the major developments of the MVS operating system was the implementation of 31-bit addressing. Prior to MVS/XA the highest virtual storage location that could be addressed was 16 megabytes, or hexadecimal *FFFFFF*. Actually, it was one byte less that 16 megabytes, because we start at zero. As applications grew larger the 24-bit architecture limitations were recognized, and 31-bit addressability was introduced. The 31-bit standard increased the amount of addressable virtual storage to 2 gigabytes. The addressing mode of a program is determined by the high order bit (bit 32 of the PSW) of the instruction address. If this bit is set to *1* the processor is running in 31-bit mode. If it is *0* then the processor is running in 24-bit mode.

8.2.2 Format of the PSW

The PSW is 64 bits in length and is actually two 32-bit words. The first 32 bits (identified as bits 0 through 31) relate to system state and mode status, but the second 32 bits (identified as bits 32 through 63) indicate the addressing mode in the first bit and the address of the next instruction in bits 33 through 63. The second word is what will interest us in most cases.

For example,

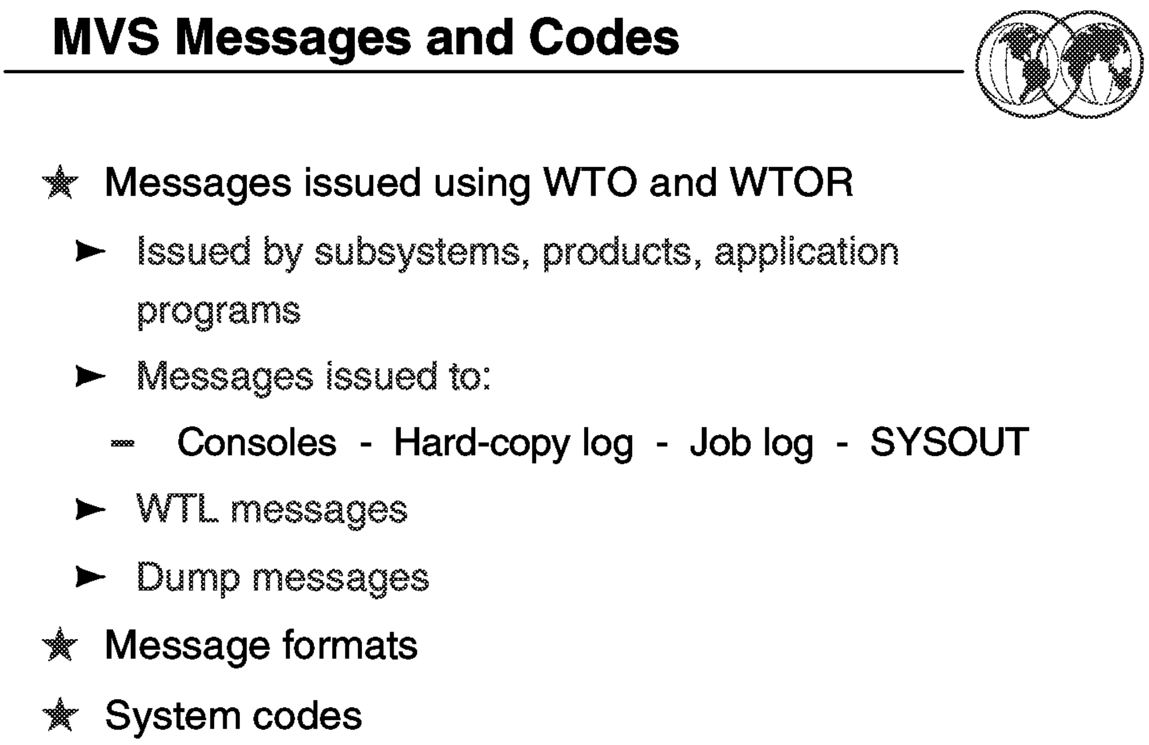
`PSW: 075C2000 82CC5BCC` Instruction length: 02

The second word of the PSW is 82CC5BCC The first number, 8, indicates that this program is executing in 31-bit mode. In other words this program runs above the 16 megabyte line. The number 8 in binary is 1000 which indicates the addressing mode bit 32 is ON. A value of zero decimal, would be binary zero, 0000, indicating that the addressing mode bit 32 was OFF which identifies that this location was below the 16 bit line, or in 24-bit mode.

The remaining data points to the next instruction to be executed. In this case, 2CC5BCC. For the sake of correctness the full address would be 02CC5BCC.

Subtracting the instruction length value, in this case, 2 from the PSW address, would result in 02CC5BCA, which would point to the failing instruction.

Later in this chapter we will learn how to use this address to locate the failing instruction.



MVS Messages and Codes

- Messages issued using WTO and WTOR
  - Issued by subsystems, products, application programs
  - Messages issued to:
    - Consoles - Hard-copy log - Job log - SYSOUT
  - WTL messages
  - Dump messages
- Message formats
- System codes

Figure 155. MVS messages and codes

8.3 MVS messages and codes

The MVS operating system issues messages from the base control program components, the job entry subsystems (JES2 and JES3), the Data Facility Product (DFP), system products, and application programs running under the system. The system issues messages in different ways and to different locations:

- Most messages are issued through WTO and WTOR macros to one of the following locations:
  - Console
  - Hard-copy log
  - Job log
  - SYSOUT data set
- Other messages are issued through the WTL macro or the LOG operator command to the system log (SYSLOG).
- Dump messages are issued through the dumping services routines and can appear in:
  - SVC dumps, stand-alone dumps, or SYSMDUMP ABEND dumps formatted by the interactive problem control system (IPCS)
  - Trace data sets formatted by the interactive problem control system (IPCS)
  - ABEND dumps or SNAP dumps produced by the dumping services

  In dump or trace data sets formatted by IPCS, the messages appear interactively on a terminal or in a printed dump.

- Some messages are issued through the Data Facility Product (DFP) access methods directly to one of the following locations:
  - Output data set
  - Display terminal

8.3.1 Message formats

The message formats are as follows:

```
id CCCnnn text
id CCCnnns text
id CCCnnnns text
id CCCnnnnns text
id CCCSnnns text
```

`id` is the reply identifier and it is optional. It appears if an operator reply is required. The operator specifies it in the reply.

`CCCnnn`, `CCCnnns`, `CCCnnnns`, `CCCnnnnns`, `CCCSnnns` is the message identifier, where: `CCC` is a prefix to identify the component, subsystem, or product that produced the message. The prefix is three characters.

`S` is the subcomponent identifier, which is an optional addition to the prefix to identify the subcomponent that produced the message. The subcomponent identifier is one character.

`nnn`, `nnnn`, `nnnnn` is a serial number that identifies the individual message. The serial number is three, four, or five decimal digits. `s` is an optional type code which can be specified as follows:

	The operator must perform a specific action.
D	The operator must choose an alternative.
E	The operator must perform action when time is available.
I	No operator action is required. Most information messages are for a programmer.
S	Severe error messages are for a programmer.
W	Processing stops until the operator performs a required action.

8.3.2 System codes

System Codes include system completion codes (or abend codes) identified by three hexadecimal digits and user completion codes identified by four decimal digits, and are usually the result of a system or an application program abnormally ending. The completion code indicates the reason for the abnormal end.

System codes also identify *wait states*. The wait state code appears in the program status word (PSW) when the operating system enters a wait state. A valid PSW for a coded wait state has one of the following general formats:

X′000A0000 8rrrrwww′

Where:

000A0000	The **A** is bits 12 through 15 (the CMWP bits).
8rrrr	A supplement code for the accompanying the wait state code appears in bits 32 through 51.

The wait state determines the size and position of the supplement code.

Usually the supplement code is a reason code. Some wait states do not provide a supplement code in the PSW.

X′000A0000 xrrxxwww′

Where:

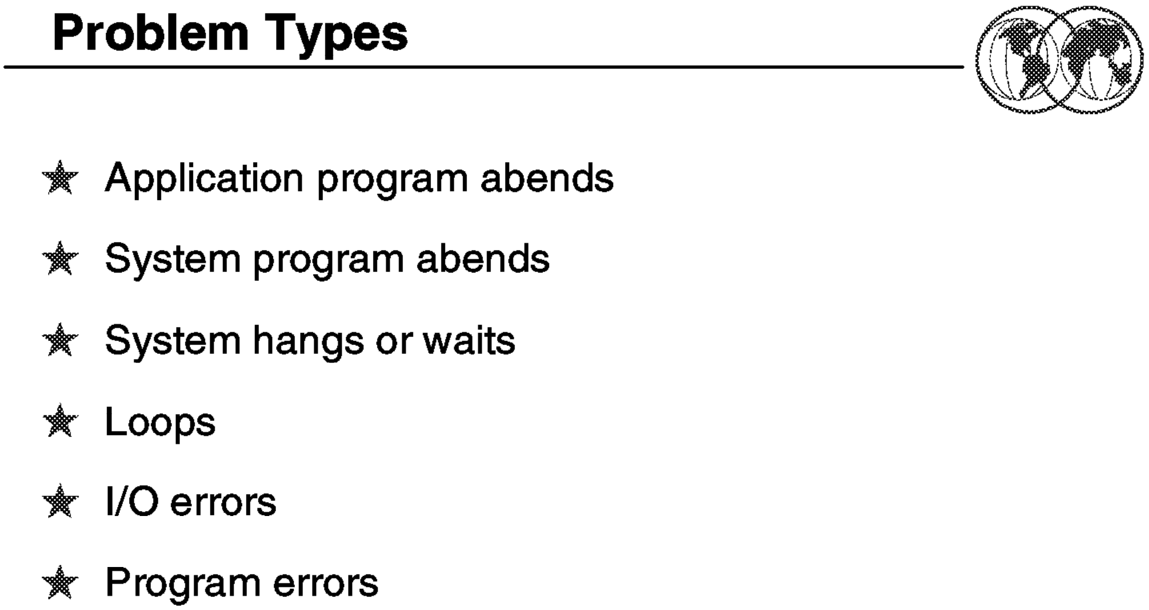
xrr Bit 32, **x** must be on (x=1), indicating 31-bit addressing mode.

rr The reason code for the accompanying wait state code which appears in bits 36 through 43.

Usually the supplement code is a reason code. Some wait states do not provide a supplement code in the PSW.

www IBM-supplied wait state code.

The IBM-supplied wait state codes are listed in *OS/390 MVS Systems Codes*, GC28-1780.



Problem Types

★ Application program abends

★ System program abends

★ System hangs or waits

★ Loops

★ I/O errors

★ Program errors

Figure 156. Problem types

8.4 Problem types

There are several problem situations that require different diagnostic methodologies, as shown in the visual.

8.4.1 Application program abends

Application program abends are always accompanied by messages in the system log (SYSLOG) and the job log indicating the abend code and usually a reason code. Many abends also generate a symptom dump in the SYSLOG and job log. A symptom dump is a system message, either message IEA995I or a numberless message, which provides some basic diagnostic information for diagnosing an abend. Often the symptom dump information can provide enough information to diagnose a problem.

The following example shows the symptom dump for an abend X′0C4′ with reason code X′4′ This symptom dump shows that:

- Active load module ABENDER is located at address X′00006FD8′.
- The failing instruction was at offset X′12′ in load module ABENDER.
- The address space identifier (ASID) for the failing task was X′000C′.

```
IEA995I SYMPTOM DUMP OUTPUT
SYSTEM COMPLETION CODE=0C4  REASON CODE=00000004
 TIME=16.44.42  SEQ=00057  CPU=0000  ASID=000C
 PSW AT TIME OF ERROR  078D0000   00006FEA  ILC 4  INTC 04
   ACTIVE LOAD MODULE=ABENDER   ADDRESS=00006FD8  OFFSET=00000012
   DATA AT PSW  00006FE4 - 00105020  30381FFF  58E0D00C
   GPR  0-3  FD000008  00005FF8  00000014  00FD6A40
   GPR  4-7  00AEC980  00AFF030  00AC4FF8  FD000000
   GPR  8-11 00AFF1B0  80AD2050  00000000  00AFF030
   GPR 12-15 40006FDE  00005FB0  80FD6A90  00006FD8
 END OF SYMPTOM DUMP
```

If the information in a symptom dump is insufficient you can capture additional dump data by including specific DD statements as discussed in the following section.

8.4.2 System program abends

Like application program abends, system program abends are usually accompanied by messages in the system log (SYSLOG) and if there is an SYS1.DUMPxx data set available at the time of the abend, and this dump code was not suppressed by the dump analysis and elimination (DAE) facility, then an SVC dump will be taken. SVC dumps will discussed later in this chapter.

8.4.3 I/O errors

I/O errors are most likely caused by a hardware failure or malfunction, and the visible symptom will be an abend, accompanied by messages in the SYSLOG which include reason codes, which can identify the type of error, and sense data, which will indicate more detailed, hardware-specific information.

I/O errors can also be the result of software conditions that create a situation where subsequent operations will appear as I/O errors. This could be the result of a corruption in a data set, or data set directory, and the rectification process may be as simple as redefining the data set.

Genuine hardware errors will generally need to be reviewed by the hardware service representative, and the diagnostic data required to analyze these problems are recorded in the system error log data set, SYS.LOGREC. IBM provide the EREP facility to enable diagnostic information to be extracted from the SYS1.LOGREC data set.

8.4.4 System wait states

The basic summation of a wait state, is the “machine is dead, and/or, will not IPL.” You will usually experience this condition during the IPL process, and the disabled wait state code will indicate the problem. The cause is often as simple as the system not being able to find some data that is crucial to the IPL process on the IPL volume. Wait codes are documented in *OS/390 MVS Systems Codes*, GC28-1780.

8.4.5 System, subsystem, and application hangs

“Hangs” are usually caused by a task, or tasks, waiting for an event that will either never happen, or an event that is taking an excessive amount of time to occur. If one of the waiting tasks is a fundamental system task, or is holding control of a resource, for example, a data set, then other tasks will queue up and wait for the required resource to become available. As more tasks enter the system they will also join the queue until the system eventually stops, or the task causing the contention is cancelled. Unfortunately, by the time the system grinds to a halt, the operating system will no longer process any operator commands, so an IPL will be the only alternative. A system hang is more specifically known as an *enabled wait* state.

8.4.6 Loops

"Loops" are caused by a program, application, or subsystem attempting to execute the same instructions over and over again. The most severe loop condition causes the task experiencing the condition to use all available CPU resources, and subsequently no other task is allowed to gain control. The only way to alleviate the problem is to cancel the problem task, or if this is unsuccessful an IPL will be required. The three types of loop conditions are:

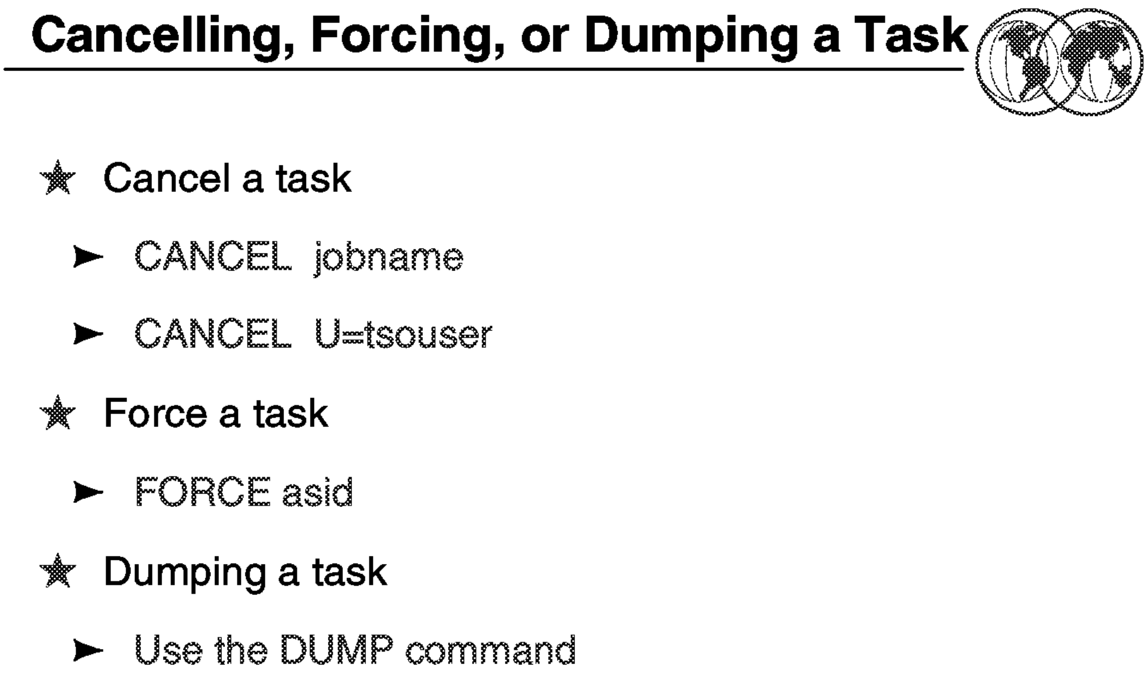
Enabled Enabled loops are usually caused by a programming error, but do not impact other jobs in the system, unless the looping task is a subsystem, which will generally impact the whole system.

Disabled Disabled loops will not allow an interrupt to be processed, and are generally identified by continuous 100 percent CPU utilization.

Spin Spin loops occur when one processor in a multiple-processor environment is unable to communicate with another processor, or is unable to access a resource being held by another processor.

8.4.7 Program errors

Program errors (error messages, bad return codes, and/or incorrect processing) require different diagnostic procedures to assist with the problem determination. Application-program-related errors are best solved in consultation with the application development team, and a combination of a system dump as well as the source code, and compile and link-edit listings can assist with the diagnosis.



Cancelling, Forcing, or Dumping a Task

- Cancel a task
  - CANCEL jobname
  - CANCEL U=tsouser
- Force a task
  - FORCE asid
- Dumping a task
  - Use the DUMP command

Figure 157. Cancelling, forcing, or dumping a task

8.5 Cancelling, forcing, or dumping a task

Cancelling a problem task can be initiated from either an MVS console or from an SDSF session running under TSO provided sufficient security privileges have been set up. The MVS console has the highest dispatching priority which allows commands to be issued at a sufficient level to handle most system loop or hang conditions. An IPL will be required if the problem task cannot be terminated using these procedures. Attempting to cancel a looping task via an SDSF session executing under TSO will often fail because the TSO session will have an insufficient dispatching priority to interrupt the loop process, but this is dependant on the severity of the looping process.

8.5.1 Cancelling a task

CANCEL can be performed as follows:

1. Issue the `CANCEL jobname` from the master console, where `jobname` is the looping task.
2. If the looping task is a TSO user, then issue, `CANCEL U=tsouser`.
3. Optionally, you might want to take a dump during the cancel. This is acheived by adding the `DUMP` option to the `CANCEL` command. For example,

   ```
   CANCEL jobname,DUMP
   ```

   It is recommended that a separate `DUMP` command be issued, and after this has been successfully processed, then CANCEL the task. This will dump according to the JCL SYSABEND, SYSUDUMP, or SYSMDUMP DD statements.

8.5.2 Forcing a task

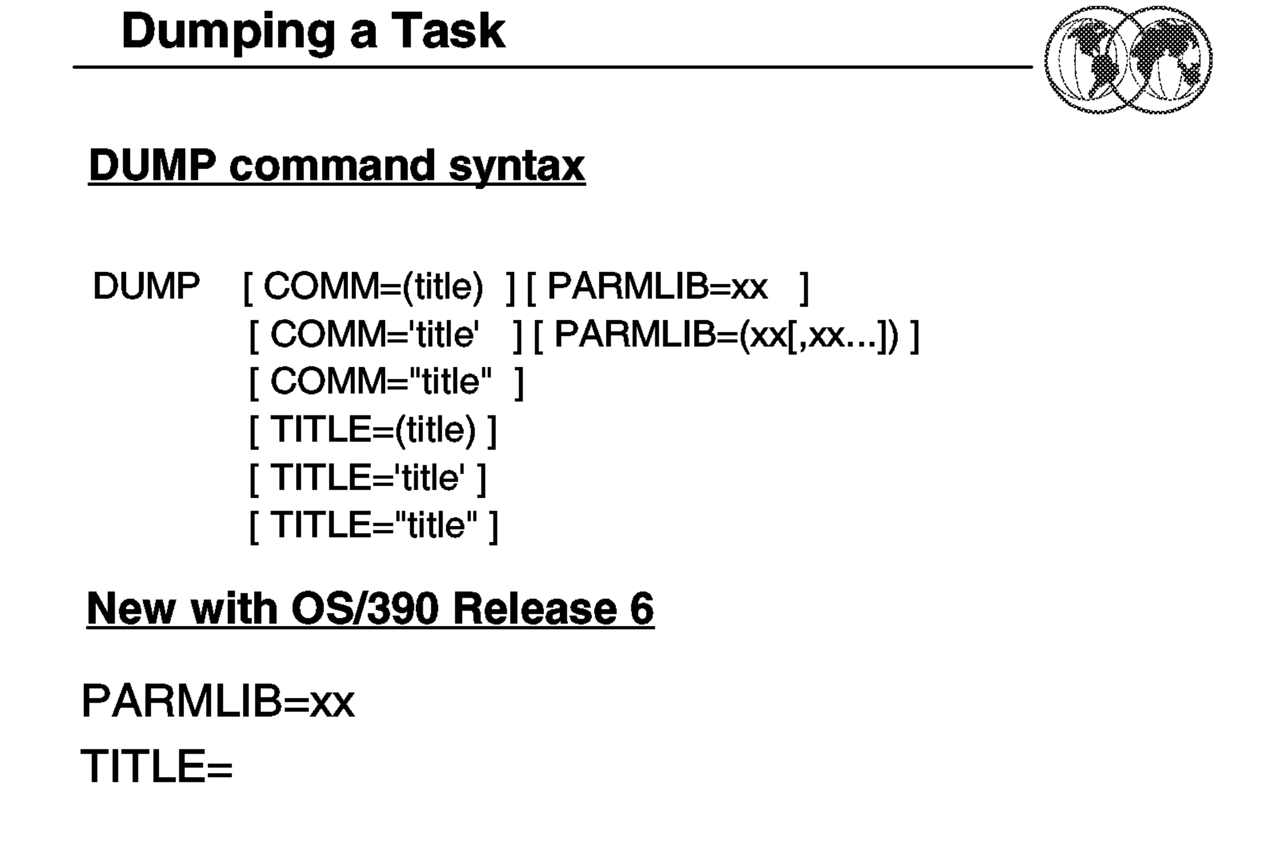
If the attempt to CANCEL a problem task (which can sometime require several CANCEL attempts, has been unsuccessful, then the next step will be to FORCE the task. FORCE is not a substitute for CANCEL. Unless you issue CANCEL first for a cancellable job, the system issues error message IEE838I. The steps to use in the process are:

1. Issue the `CANCEL nnn command`, making several attempts if necessary.
2. Use the `DUMP` command—if you want a dump produced. Respond to the prompt for parameters with the jobname or ASID of the "stuck" job, as well as ASID(1)=MASTER.
3. Issue the `FORCE nnn,ARM` command for non-cancellable procedures.
4. Issue the `FORCE nnn` command only when the previous steps fail.

Warning

Never use the `FORCE` command without understanding that:

- After issuing `FORCE`, you might have to re-IPL.
- If you issue `FORCE` for a job in execution or for a time-sharing user, the system deletes the affected address space and and severely limits recovery unless you use the ARM parameter. (ARM is described below.)
- If you need a dump, you must issue a `DUMP` command before you issue `FORCE`. After you've issued a `FORCE` command it is usually *not possible* to get a dump of the failing address space.
- If your system was part of a global resource serialization ring (GRS=START, GRS=JOIN or GRS=TRYJOIN was specified at IPL) but has been quiesced (by entering the `VARY GRS(system name),QUIESCE command`), FORCE processing might not complete immediately. The system suspends termination of all address spaces holding global resources until the quiesced system rejoins the ring or is purged from the ring. Use a `DISPLAY GRS` command to determine GRS status.
- When you use the `FORCE` command to end the availability manager (AVM) address space, the operator must restart that address space by issuing the command `START AVM,SUB=MSTR`.
- You can enter `FORCE` only from a console with master authority.



Dumping a Task

DUMP command syntax

```
DUMP    [ COMM=(title)  ] [ PARMLIB=xx   ]
        [ COMM='title'  ] [ PARMLIB=(xx[,xx...]) ]
        [ COMM="title"  ]
        [ TITLE=(title) ]
        [ TITLE='title' ]
        [ TITLE="title" ]
```

New with OS/390 Release 6

PARMLIB=xx

TITLE=

Figure 158. Dumping a task

8.5.3 Dumping a task

The DUMP command requests a system dump (SVC dump) of virtual storage. The data set may be either a pre-allocated dump data set named SYS1.DUMPxx, or an automatically allocated dump data set named according to an installation-specified pattern. You should request only one dump at a time on one system. A system writes only one SVC dump at a time, so it does not save time to make several requests at once.

The format of the DUMP command is shown in the visual with several new keywords becoming available with OS/390 Release 6.

The title is the name (1 to 100 characters) you want the dump to have. This title becomes the first record in the dump data set. COMM= and TITLE= are synonyms.

You can also use the parmlib parameter as follows:

```
DUMP COMM=(..........),PARMLIB=(xx)
```

The PARMLIB= parameter allows you to provide lengthy DUMP command specifications through a parmlib member. The two alphanumeric characters xx indicate the IEADMCxx member of SYS1.PARMLIB that contains the DUMP command specification. The syntax of a DUMP command specified within the IEADMCxx members of SYS1.PARMLIB is identical to that specified on the DUMP command through writes to operator with reply (WTORs).

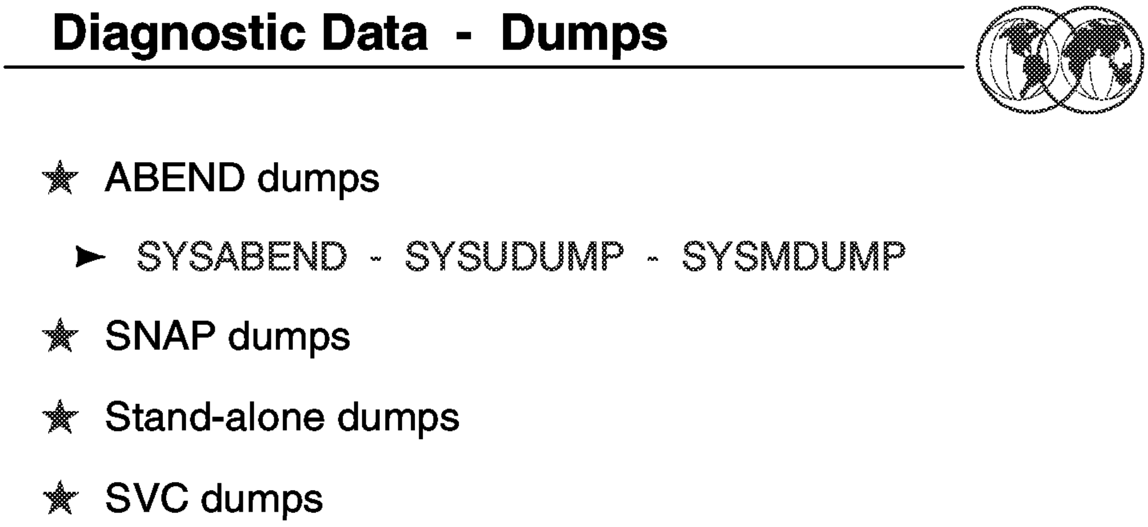
In response to the DUMP command, the system prompts you with the following message for the dump options you want to specify:

```
* id IEE094D SPECIFY OPERAND(S) FOR DUMP COMMAND
```

The responses to the IEE094D message are any one of the following:

```
R id,U
R id,ASID=n
R id,JOBNAME=jobname
R id,TSONAME=name
R id,DSPNAME=dspname-entry
         -  which is used for data spaces.
```

Note: The WTOR is not issued when the PARMLIB= parameter is specified.



Diagnostic Data - Dumps

- ABEND dumps
  - SYSABEND - SYSUDUMP - SYSMDUMP
- SNAP dumps
- Stand-alone dumps
- SVC dumps

Figure 159. Diagnostic data - dumps

8.6 Diagnostic data - dumps

There are different types of dumps and traces that can be used to analyze problems. The dump types and the procedures that can be used to initiate these processes are shown on the visual.

Dumps could best be described as a “SNAP shot” of the system at the time a failure is detected by the operating system or application, or at the time the system is dumped by the operator via the DUMP command or the stand-alone dump procedure.

8.6.1 ABEND dumps

The system can produce three types of ABEND dumps, as follows:

SYSABEND The largest of the ABEND dumps, containing a summary dump for the failing program plus many other areas useful for analyzing processing in the failing program. This dump is formatted.

SYSMDUMP Contains a summary dump for the failing program, plus some system data for the failing task. SYSMDUMP dumps are the only ABEND dumps that are unformatted and must be formatted with IPCS.

SYSUDUMP The smallest of the ABEND dumps, containing data and areas only about the failing program. This dump is formatted.

You can obtain SYSABEND, SYSUDUMP, and SYSMDUMP dumps by specifying the correct DD statement in your JCL:

- SYSABEND dumps are formatted as they are created and can be directed to either DASD, TAPE, or SYSOUT.

```
//SYSABEND    DD SYSOUT=*
```

- SYSUDUMP dumps are formatted as they are created and can be directed to either DASD, TAPE, or SYSOUT.

```
//SYSUDUMP    DD SYSOUT=*
```

- SYSMDUMP dumps are unformatted and must be analyzed using the Interactive Problem Control System (IPCS). These data sets must reside on either DASD or TAPE.

```
//SYSMDUMP    DD DSN=MY.SYSMDUMP,DISP=(,CATLG),UNIT=DISK,
//               SPACE=(CYL,(50,20),RLSE),
//               LRECL=4160,BLKSIZE=4160
```

ABEND dumps can be suppressed using the SLIP command in member IEASLPxx in SYS1.PARMLIB. These commands used to reside in member IEACMDxx in SYS1.PARMLIB but it is recommended that you move any SLIP commands from IEACMDxx to IEASLPxx to avoid restrictions fornd in other parmlib members. For example,

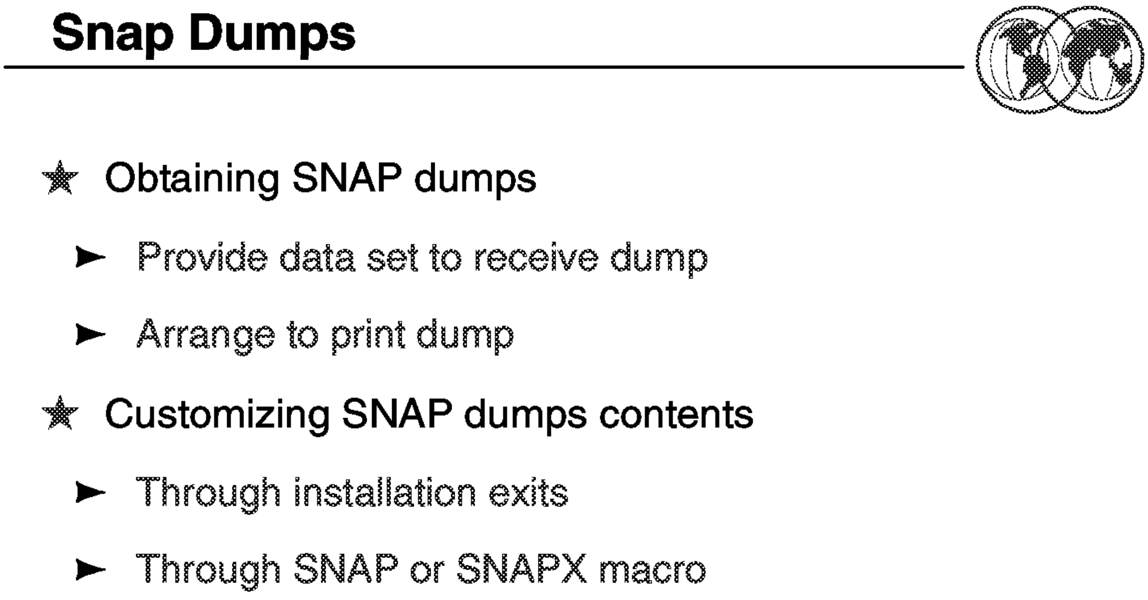
- IEASLPxx supports multiple-line commands; IEACMD00 does not.
- IEASLPxx does not require any special command syntax; IEACMD00 does.

Figure 160 shows SLIP commands is SYS1.PARMLIB member IEASLP00.

SLIP commands in SYS1.PARMLIB member IEASLP00

```
SLIP SET,C=013,ID=X013,A=NOSVCD,J=JES2,END
SLIP SET,C=028,ID=X028,A=NOSVCD,END
SLIP SET,C=47B,DATA=(15R,EQ,0,OR,15R,EQ,8),ID=X47B,A=NODUMP,END
SLIP SET,C=058,DATA=(15R,EQ,4,OR,15R,EQ,8,OR,15R,EQ,C,OR,15R,EQ,10,OR,
   15R,EQ,2C,OR,15R,EQ,30,OR,15R,EQ,3C),ID=X058,A=NODUMP,END
SLIP SET,C=0E7,ID=X0E7,A=NOSVCD,END
SLIP SET,C=0F3,ID=X0F3,A=NODUMP,END
SLIP SET,C=13E,ID=X13E,A=NODUMP,END
SLIP SET,C=222,ID=X222,A=NODUMP,END
SLIP SET,C=322,ID=X322,A=NODUMP,END
SLIP SET,C=33E,ID=X33E,A=NODUMP,END
SLIP SET,C=422,ID=X422,A=NODUMP,END
SLIP SET,C=622,ID=X622,A=NODUMP,END
SLIP SET,C=804,ID=X804,A=(NOSVCD,NOSYSU),END
SLIP SET,C=806,ID=X806,A=(NOSVCD,NOSYSU),END
SLIP SET,C=80A,ID=X80A,A=(NOSVCD,NOSYSU),END
SLIP SET,C=9FB,ID=X9FB,A=NOSVCD,J=JES3,END
SLIP SET,C=B37,ID=XB37,A=(NOSVCD,NOSYSU),END
SLIP SET,C=D37,ID=XD37,A=(NOSVCD,NOSYSU),END
SLIP SET,C=E37,ID=XE37,A=(NOSVCD,NOSYSU),END
SLIP SET,C=EC6,RE=0000FFXX,ID=XEC6,A=NODUMP,END
SLIP SET,C=EC6,RE=0000FDXX,ID=XXC6,A=NOSVCD,END
```

Figure 160. SLIP commands in SYS1.PARMLIB member IEASLP00



Snap Dumps

- Obtaining SNAP dumps
  - Provide data set to receive dump
  - Arrange to print dump
- Customizing SNAP dumps contents
  - Through installation exits
  - Through SNAP or SNAPX macro

Figure 161. SNAP dumps

8.6.2 SNAP dumps

Use a SNAP dump when testing a problem program. A SNAP dump shows one or more areas of virtual storage that a program, while running, requests the system to dump. A series of SNAP dumps can show an area at different stages in order to picture a program′s processing, dumping one or more fields repeatedly to let the programmer check intermediate steps in calculations. SNAP dumps are preformatted, you cannot use IPCS to format them.

Note: A SNAP dump is written while a program runs, rather than during abnormal end.

8.6.2.1 Obtaining a SNAP dump

Obtain a SNAP dump by taking the following steps:

1. Code a DD statement in the JCL for the job step that runs the problem program to be dumped with a ddname other than SYSUDUMP, SYSABEND, SYSMDUMP, or another restricted ddname. The statement can specify that the output of the SNAP dump should be written to one of the following:
   - Direct access storage device (DASD). For example,

     ```
     //SNAP1 DD DSN=MY.SNAP.DUMP,DISP=(OLD)
     ```

   - Printer. Note that a printer is not recommended because the printer cannot be used for anything else while the job step is running, whether a dump is written or not.
   - SYSOUT. SNAP dumps usually use SYSOUT. For example,

     ```
     //SNAP1 DD SYSOUT=X
     ```

- Tape. For example,

  ```
  //SNAP1 DD DSN=SNAP.TO.TAPE,UNIT=TAPE,DISP=(OLD)
  ```

2. In the problem program:

   a. Specify a data control block (DCB) for the data set to receive the dump. For a standard dump, which has 120 characters per line, the DCB must specify:

   ```
    or 1632
       DSORG=PS
       LRECL=125
       MACRF=(W)
       RECFM=VBA
   ```

   For a high-density dump, which has 204 characters per line and will be printed on an APA 3800 printer, the DCB must specify:

   ```
   BLKSIZE=1470 or 2724
   DSORG=PS
   LRECL=209
   MACRF=(W)
   RECFM=VBA
   ```

   b. Code an OPEN macro to open the DCB.

   Before you issue the SNAP or SNAPX macro, you must open the DCB that you designate on the DCB parameter, and ensure that the DCB is not closed until the macro returns control. To open the DCB, issue the DCB macro with the following parameters, and issue an OPEN macro for the data set:

   ```
   DSORG=PS,RECFM=VBA,MACRF=(W),BLKSIZE=nnn,LRECL=xxx,
   and DDNAME=any name but SYSABEND, SYSMDUMP or SYSUDUMP
   ```

   If the system loader processes the program, the program must close the DCB after the last SNAP or SNAPX macro is issued.

   c. Code a SNAP or SNAPX assembler macro to request the dump. We recommend the use of the SNAPX macro as this allows for programs running in Access-Register (AR) mode to cause the macro to generate larger parameter lists. In the following example, the SNAPX macro requests a dump of a storage area, with the DCB address in register 3, a dump identifier of 245, the storage area′s starting address in register 4, and the ending address in register 5:

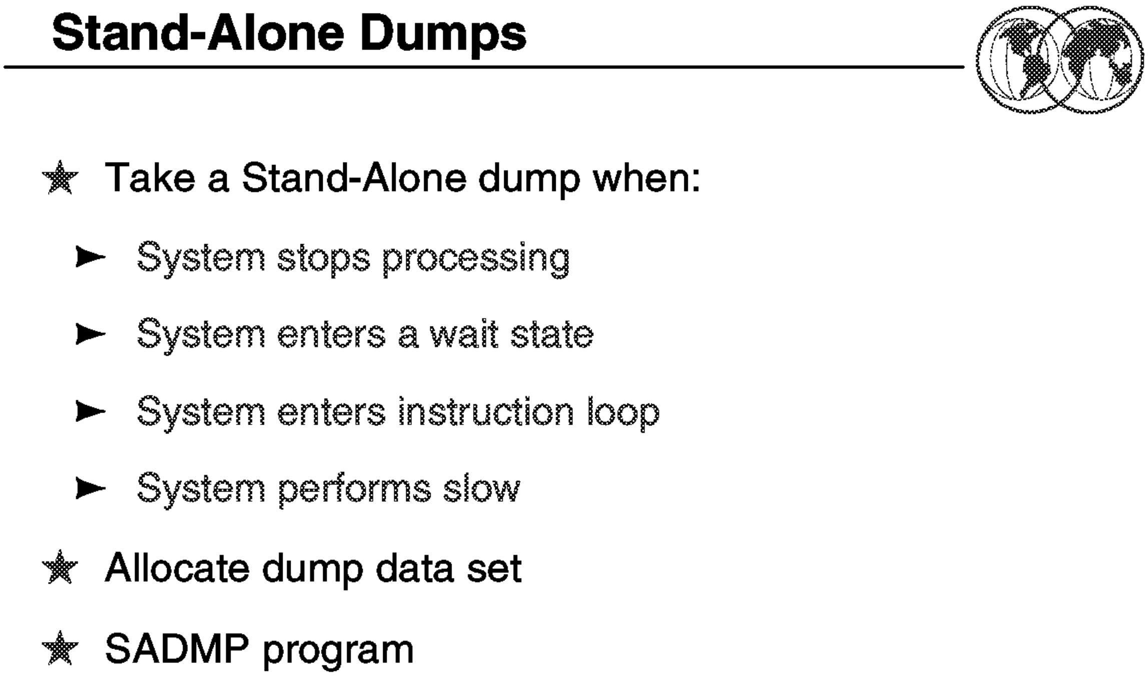
   ```
         SNAPX  DCB=(3),ID=245,STORAGE=((4),(5))
   ```

   Repeat this macro in the program as many times as wanted, changing the dump identifier for a unique dump. The system writes all the dumps that specify the same DCB to the same data set.

   d. Close the DCB with a CLOSE assembler macro.

8.6.2.2 Customizing SNAP dumps

An installation can customize the contents of SNAP dumps through the IEAVADFM or IEAVADUS installation exits. IEAVADFM is a list of installation routines to be run and IEAVADUS is one installation routine. The installation exit routine runs during control block formatting of a dump when the CB option is specified on the SNAP or SNAPX macro. The routine can format control blocks and send them to the data set for the dump. See *OS/390 MVS Installation Exits*, SC28-1753, for more information.



Stand-Alone Dumps

- Take a Stand-Alone dump when:
  - System stops processing
  - System enters a wait state
  - System enters instruction loop
  - System performs slow
- Allocate dump data set
- SADMP program

Figure 162. Stand-alone dumps

8.6.3 Stand-alone dumps

Stand-alone dumps are not produced by OS/390 but by a program called SADMP which is IPLed in place of OS/390. When to use a stand-alone dump is shown on the visual.

These dumps show central storage and some paged-out virtual storage occupied by the system or stand-alone dump program that failed. Stand-alone dumps can be analyzed using IPCS.

The term stand-alone means that the dump is performed separately from normal system operations and does not require the system to be in a condition for normal operation.

8.6.3.1 Allocating the stand-alone dump data set

In the SYS1.SAMPLIB data set use the AMDSADDD REXX utility to allocate and initialize the SADMP dump data sets. You can EXEC this REXX utility from the ISPF data set utility option **3.4**, and either VIEW (**V**), BROWSE (**B**) or EDIT (**E**) the SYS1.SAMPLIB data set. Issue the EXEC command next to member AMDSADDD. For example:

```
EXEC_____ AMDSADDD
```

Alternatively, issue the following command from the ISPF option line.

```
TSO EXEC 'SYS1.SAMPLIB(AMDSADDD)'
```

This utility will prompt you as follows:

```
What function do you want?
Please nter DEFINE if you want to allocate a new dump data set
Please enter CLEAR if you want to clear an existing dump data set
Please enter REALLOC if you want to reallocate and clear an existing
       dump data set
Please enter QUIT if you want to leave this procedure
define

 Please enter VOLSER or VOLSER(dump_dataset_name)
SYS001
 Please enter the device type for the dump data set
 Device type choices are 3380 or 3390 or 9345
3390
 Please enter the number of cylinders
300
 Do you want the dump data set to be cataloged?
 Please respond Y or N
Y
 TIME-08:59:31 AM. CPU-00:00:03 SERVICE-549023 SESSION-01:18:42 APRIL 9,

 Initializing output dump data set with a null record:
 Dump data set has been successfully initialized

 Results of the DEFINE request:

   Dump data set Name   : SYS1.SADMP
   Volume               : SYS001
   Device Type          : 3390
   Allocated Amount     : 3

 ***
```

8.6.3.2 SADMP program

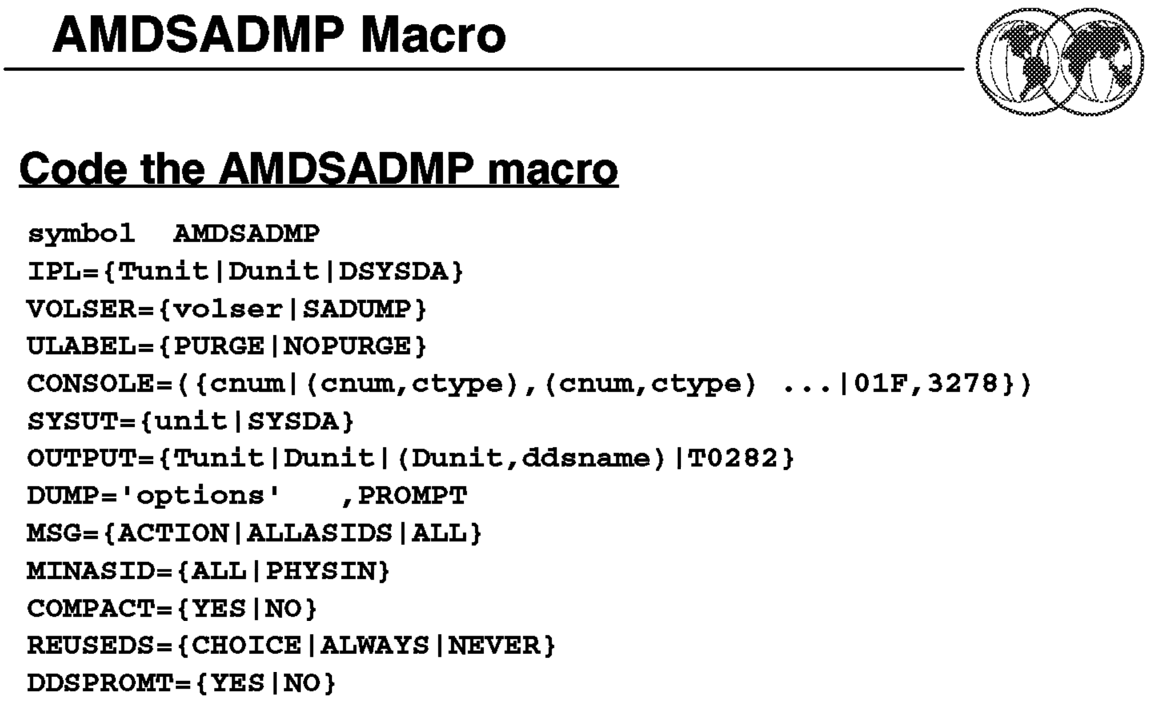
The SADMP program produces a high-speed, unformatted dump of central storage and parts of paged-out virtual storage on a tape device or a direct access storage device (DASD). The SADMP program, which you create must reside on a storage device that can be used to IPL.

You must create the SADMP program by using the AMDSADM macro to produce the following:

- A SADMP program that resides on DASD with output directed to a tape volume or to a DASD dump data set
- A SADMP program that resides on tape, with output directed to a tape volume or to a DASD dump data set

Create the SADMP program as follows:

```
//SADMPGEN JOB  MSGLEVEL=(1,1)
//OSG      EXEC PGM=AMDSAOSG
//SYSLIB   DD   DSN=SYS1.MACLIB,DISP=SHR
//         DD   DSN=SYS1.MODGEN,DISP=SHR
//DPLTEXT  DD   DSN=SYS1.NUCLEUS(AMDSADPL),DISP=SHR
//IPLTEXT  DD   DSN=SYS1.NUCLEUS(AMDSAIPD),DISP=SHR
//PGETEXT  DD   DSN=SYS1.NUCLEUS(AMDSAPGE),DISP=SHR
//GENPRINT DD   DSN=SADMP.LIST,DISP=OLD
//GENPARMS DD   *
         AMDSADMP IPL=DSYSDA,VOLSER=SPOOL2,                            X
               CONSOLE=(1A0,3277)
         END
/*
```



AMDSADMP Macro

Code the AMDSADMP macro

```
symbol  AMDSADMP
IPL={Tunit|Dunit|DSYSDA}
VOLSER={volser|SADUMP}
ULABEL={PURGE|NOPURGE}
CONSOLE=({cnum|(cnum,ctype),(cnum,ctype) ...|01F,3278})
SYSUT={unit|SYSDA}
OUTPUT={Tunit|Dunit|(Dunit,ddsname)|T0282}
DUMP='options'   ,PROMPT
MSG={ACTION|ALLASIDS|ALL}
MINASID={ALL|PHYSIN}
COMPACT={YES|NO}
REUSEDS={CHOICE|ALWAYS|NEVER}
DDSPROMT={YES|NO}
```

Figure 163. AMDSADMP macro

8.6.3.3 AMDSADMP macro

AMDSADMP processing does not allocate the data set or check to see that a valid MVS data set name has been provided. Therefore, you should insure that:

- The AMDSADDD REXX utility is used to allocate and initialize the same data set name specified on the OUTPUT= keyword.
- The data set name specified should be fully qualified (without quotes).
- The necessary data set management steps are taken so that the SADMP dump data sets will not be placed into a migrated state or moved to a different volume.
- Alphabetic characters appearing in the dump data set name should be specified as capital letters.

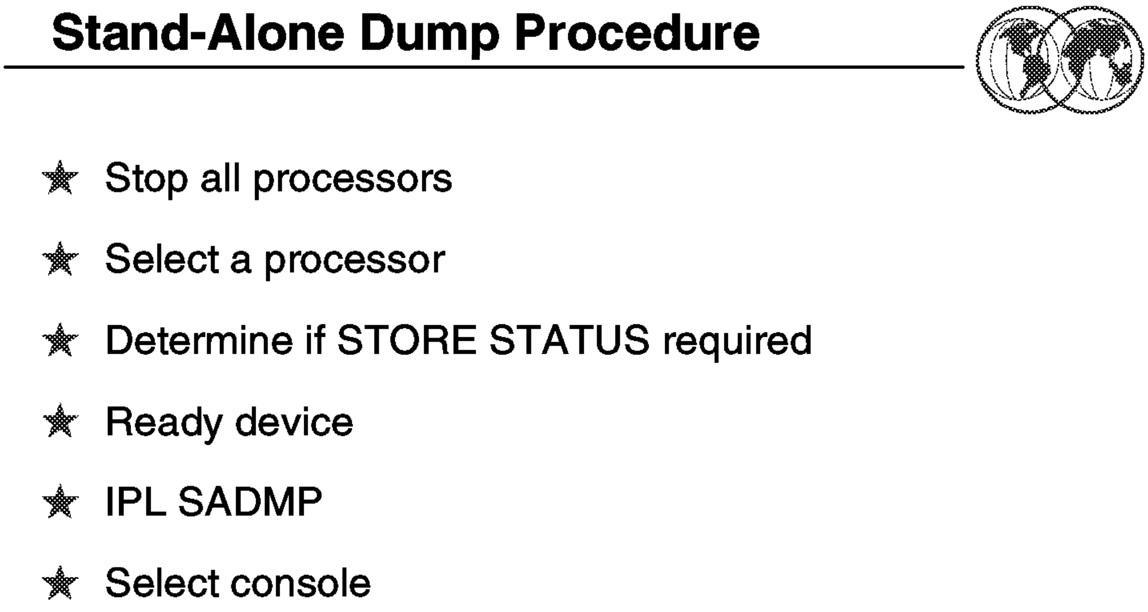
If the default DASD device is to be used and no dump data set name is provided, the SADMP program will assume that the default dump data set name is SYS1.SADMP if the DDSPROMPT=NO parameter was also specified. Otherwise, if DDSPROMPT=YES was specified, the SADMP program will prompt the operator at run-time for a dump data set name to use.

- At run-time, only a null response to message AMD001A will cause the SADMP program to use the default device and/or dump data set name.
- Do not place a data set that is intended to contain a stand-alone dump on a volume that also contains a page or swap data set that the stand-alone dump program may need to dump. When SADMP initializes a page or swap volume for virtual dump processing, it checks to see if the output dump data set also exists on this volume. If it does, the SADMP program issues message AMD100I and does not retrieve any data from page or swap data sets on this volume. Thus, the

dump may not contain all of the data that you requested. This lack of data may impair subsequent diagnosis.

- You cannot direct output to the SADMP residence volume.

See D.1, “ADMSADMP macro parameters” on page 377 for a description of the parameters shown on the visual.



Stand-Alone Dump Procedure

- Stop all processors
- Select a processor
- Determine if STORE STATUS required
- Ready device
- IPL SADMP
- Select console

Figure 164. Stand-alone dump procedure

8.6.3.4 Stand-alone dump procedure

Use the following procedure to initialize the SADMP program and dump storage:

1. Stop all processors. Do not clear storage.

2. Select a processor that was online when the system was stopped.

3. If the processor provides a function to IPL a stand-alone dump without performing a manual STORE STATUS, use this function to IPL SADMP. If you do not use such a function, perform a STORE STATUS before IPLing stand-alone dump. If the operator does not store status, virtual storage is not dumped.

   The hardware store-status facility stores the current program status word (PSW), current registers, the processor timer, and the clock comparator into the unprefixed prefix save area (PSA). This PSA is the one used before the nucleus initialization program (NIP) initialized the prefix register.

   If you IPL the stand-alone dump program from the hardware console, it is not necessary to perform the STORE STATUS operation. Status is automatically stored when stand-alone dump is invoked from the hardware console and automatic store status is on.

   If the operator does not issue the STORE STATUS instruction before IPLing a stand-alone dump, the message "ONLY GENERAL PURPOSE REGS VALID" might appear on the formatted dump. The PSW, control registers, and so on, are not included in the dump.

   Note: Do not use the LOAD CLEAR option. Using the LOAD CLEAR option erases main storage, which means that you will not be able to diagnose the failure properly.

4. Make the residence device ready. If it is a tape, mount the volume on a device attached to the selected processor and ensure that the file-protect ring is in place. If it is a DASD volume, ensure that it is write-enabled.

5. IPL SADMP

   SADMP does not communicate with the operator console. Instead, SADMP loads an enabled wait PSW with wait reason code X′3E0000′. The IPLing of the stand-alone dump program causes absolute storage (X′0′ through X′18′ and storage beginning at X′110′) to be overlaid with CCWs. You should be aware of this and not consider it as a low storage overlay.

   Note: SADMP uses the PSW to communicate with the operator or systems programmer.

   SADMP waits for a console I/O interrupt or an external interrupt.

6. Select the system console or an operator console with a device address that is in the console list that you specified at SADMP generation time (in the CONSOLE keyword of AMDSADMP). At SADMP run time, the operator can choose either a console specified with the CONSOLE= keyword or the system console to control SADMP operation. If an operator console is chosen, press Attention or Enter on that console. (On some consoles, you might have to press Reset first.) This causes an interruption that informs SADMP of the console′s address. Message AMD001A appears on the console.

   a. Make an output device ready. When you dump to devices that have both real and virtual addresses (for example, dumping a VM system), specify only the real address to the SADMP program. If you are dumping to tape, ensure that the tape cartridge is write-enabled. If you are dumping to DASD, ensure that the DASD data set has been initialized using the AMDSADDD REXX utility.

   b. Reply with the device number for the output device. If you are dumping to a DASD device and DDSPROMPT=YES was specified on the AMDSADMP macro, message AMD002A is issued to prompt the operator for a dump data set. If DDSPROMPT=NO was specified, message AMD002A is not issued and the SADMP program assumes that the dump data set name is SYS1.SADMP.

Note: Pressing Enter in response to message AMD001A will cause the SADMP program to use the default device specified on the OUTPUT= keyword of the AMDSADMP macro. If the default device is a DASD device, then pressing the Enter key in response to message AMD001A will cause the SADMP program to use both the default device and the dump data set specified on the OUTPUT= keyword of the AMDSADMP macro. If no dump data set name was provided on the OUTPUT= keyword and the DDSPROMPT=YES keyword was specified, message AMD002A is issued to prompt the operator for a dump data set. If DDSPROMPT=NO was specified, then the SADMP program assumes that the dump data set name is SYS1.SADMP.

If you reply with the device number of an attached device that is not of the required device type, or if the device causes certain types of I/O errors, SADMP might load a disabled wait PSW. When this occurs, use procedure B to restart SADMP.

8.6.3.5 SADMP processing

SADMP prompts you, with message AMD011A, for a dump title. When no console is available, run SADMP without a console.

- Ready the default output device that was specified on the OUTPUT parameter on the AMDSADMP macro. For tapes, ensure that the tape cartridge is write-enabled. For DASD, ensure that the dump data set has been initialized using the AMDSADDD REXX utility.
- Enter an external interruption on the processor that SADMP was IPLed from. SADMP proceeds using the default output device and/or the default dump data set. No messages appear on any consoles; SADMP uses PSW wait reason codes to communicate to the operator.

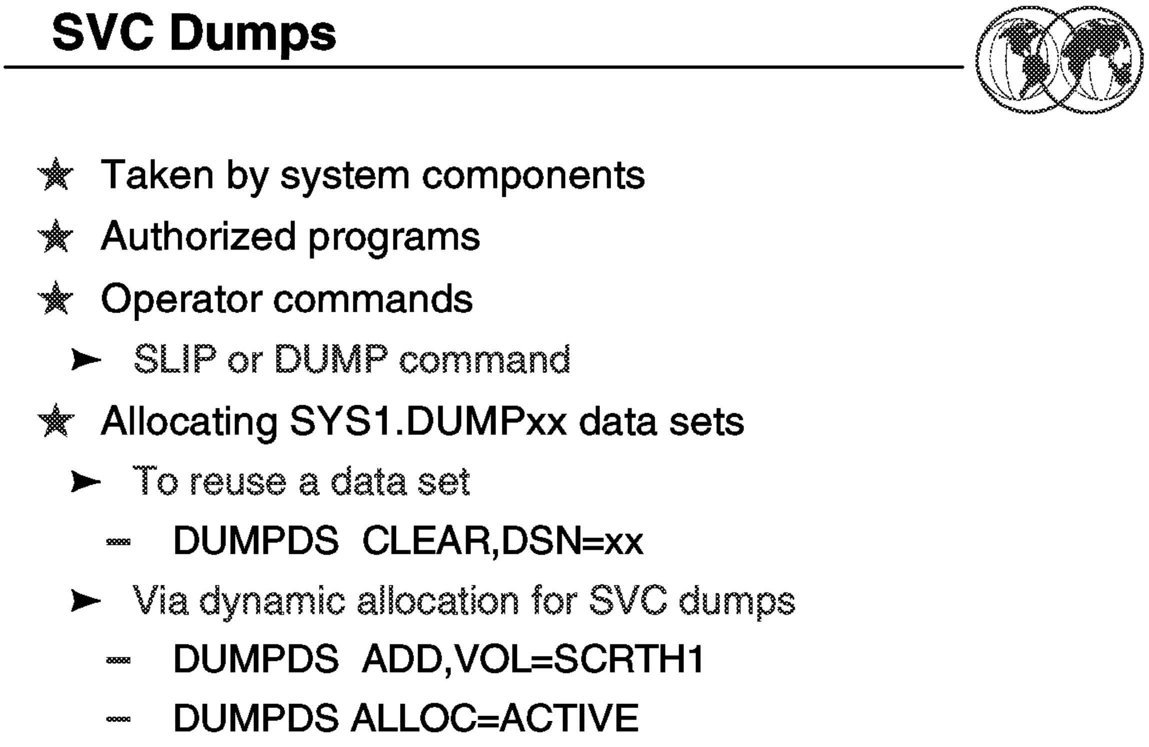
When SADMP begins and finishes dumping central storage, it issues message AMD005I to display the status of the dump. SADMP may end at this step.

When SADMP begins dumping real storage it issues message AMD005I. Message AMD095I is issued every 30 seconds to indicate the progress of the dump. Message AMD005I will be issued as specific portions of real storage have been dumped, as well as upon completion of the real dump. SADMP may end at this step.

If you specified PROMPT on the AMDSADMP macro, SADMP prompts you for additional storage that you want dumped by issuing message AMD059D.

SADMP dumps instruction trace data, paged-out virtual storage, the SADMP message log, and issues message AMD095I every 30 seconds to indicate the progress of the dump.

When SADMP completes processing, SADMP unloads the tape, if there is one, and enters a wait reason code X′410000′.



SVC Dumps

- Taken by system components
- Authorized programs
- Operator commands
  - SLIP or DUMP command
- Allocating SYS1.DUMPxx data sets
  - To reuse a data set
    - DUMPDS CLEAR,DSN=xx
  - Via dynamic allocation for SVC dumps
    - DUMPDS ADD,VOL=SCRTH1
    - DUMPDS ALLOC=ACTIVE

Figure 165. SVC dumps

8.6.4 SVC dumps

SVC dumps can be used in different ways:

- Most commonly, a system component requests an SVC dump when an unexpected system error occurs, but the system can continue processing.
- An authorized program or the operator can also request an SVC dump (by using the SLIP or DUMP command) when they need diagnostic data to solve a problem.

SVC dumps contain a summary dump, control blocks, and other system code, but the exact areas dumped depend on whether the dump was requested by a macro, command, or SLIP trap. SVC dumps can be analyzed using IPCS.

SVC dump processing stores data in dump data sets that you pre-allocate manually, or that the system allocates automatically, as needed. You can also use pre-allocated dump data sets as a back up in case the system is not able to allocate a data set automatically. To prepare your installation to receive SVC dumps, you need to provide SYS1.DUMPxx data sets. These data sets will hold the SVC dump information for later review and analysis. This section describes how to set up the SVC dump data sets, including:

8.6.4.1 Allocating SYS1.DUMPxx data sets

Allocate SYS1.DUMPxx data sets using the following requirements:

- Name the data set SYS1.DUMPxx, where xx is a decimal number of 00 through 99.
- Select a device with a track size of 4160 bytes. The system writes the dump in blocked records of 4160 bytes.
- Initialize with an end of file (EOF) record as the first record.
- Allocate the data set before requesting a dump. Allocation requirements are:

  UNIT A permanently resident volume on a direct access device.

  DISP Catalog the data set (CATLG). Do not specify SHR.

  VOLUME Place the data set on only one volume. Allocating the dump data set on the same volume as the page data set could cause contention problems during dumping, as pages for the dumped address space are read from the page data set and written to the dump data set.

  SPACE An installation must consider the size of the page data set that will contain the dump data. The data set must be large enough to hold the amount of data as defined by the MAXSPACE parameter on the CHNGDUMP command, VIO pages, and pageable private area pages. SVC dump processing improves service by allowing secondary extents to be specified when large dump data sets are too large for the amount of DASD previously allocated. An installation can protect itself against truncated dumps by specifying secondary extents and by leaving sufficient space on volumes to allow for the expansion of the dump data sets. For the SPACE keyword, you can specify CONTIG to make reading and writing the data set faster. Request enough space in the primary extent to hold the smallest SVC dump expected. Request enough space in the secondary extent so that the primary plus the secondary extents can hold the largest SVC dump. The actual size of the dump depends on the dump options in effect when the system writes the dump.

  Note: Approximately 250 cylinders will be sufficient for most SVC dump requirements.

  The system writes only one dump in each SYS1.DUMPxx data set. Before the data set can be used for another dump it can be cleared by using the DUMPDS command with the CLEAR keyword. The format if the command is:

  ```
  DUMPDS CLEAR,DSN=xx
  ```

  Where xx is the SYS1.DUMPxx identifier. You can abbreviate the DUMPDS command to DD, for example:

  ```
  DD CLEAR,DSN=01
  ```

8.6.4.2 Dynamic allocation of SVC dump data sets

SVC dump processing supports automatic allocation of dump data sets at the time the system writes the dump to DASD. The dump can be allocated from a set of DASD volumes or SMS classes. When the system captures a dump, it allocates a data set of the correct size from the resources you specify. If automatic allocation fails, pre-allocated dump data sets are used. If no pre-allocated SYS1.DUMPnn data sets are available, message IEA793A is issued, and the dump remains in virtual storage. SVC dump periodically retries both automatic allocation and writing to a pre-allocated dump data set until successful or until the captured dump is deleted either by operator intervention or by the expiration of the CHNGDUMP MSGTIME parameter governing message IEA793A.

You can specify the command instructions to enable or disable automatic allocation either in the COMMNDxx parmlib member, to take effect at IPL, or from the operator console at any time after the

IPL, to dynamically modify automatic allocation settings. The DUMPDS command provides the following flexibility:

- Activate automatic allocation of dump data sets
- Add or delete allocation resources
- Direct automatic allocation to SMS or non-SMS managed storage
- Deactivate automatic allocation of dump data sets
- Reactivate automatic allocation of dump data sets
- Change the dump data set naming convention

Automatic allocation can be set up using the following steps:

- Set up allocation authority
- Establish a name pattern for the data sets
- Define resources for storing the data sets
- Activate automatic allocation

Once active, allocation to SMS classes and DASD volumes is done starting from the first resource you added with the DUMPDS ADD command until unsuccessful, then the next resource is used. If you have defined both DASD volumes and SMS classes, SMS classes are used first. Allocation to DASD volumes is not multivolume or striped, whereas allocation to SMS classes can be multivolume or striped, depending on how the storage class is set up by the installation.

The steps to initiate automatic dump data set allocation are:

- Associate the DUMPSRV address space with a user ID.

- Authorize DUMPSRV′s user ID to create new dump data sets.

- Set up your installation data set name pattern using the DUMPDS command:

  ```
  DUMPDS NAME=&SYSNAME;.&JOBNAME;.Y&YR4;M&MON;.D&DAY;T&HR;&MIN;.S&SEQ;
  ```

- Add dump data set resources that can be used by the automatic allocation function:

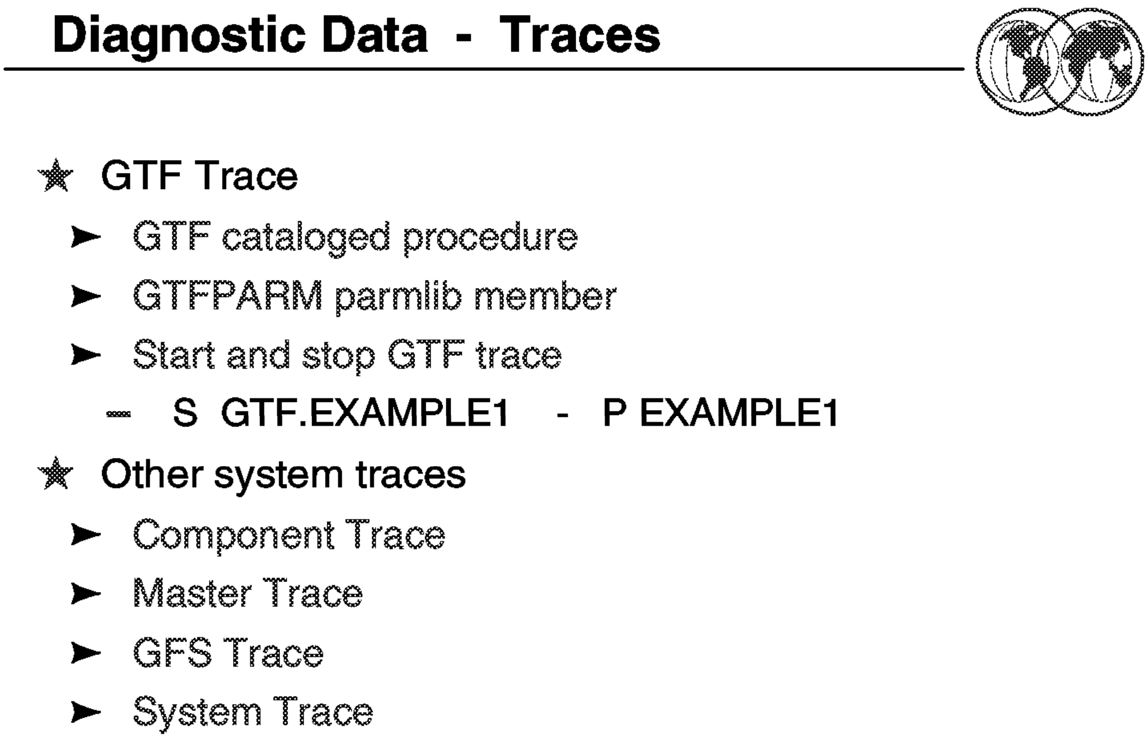
  ```
      DUMPDS ADD,VOL=(SCRTH1,HSM111)
      DUMPDS ADD,SMS=(DUMPDA)
  ```

- Activate automatic dump data set allocation using the DUMPDS command:

  ```
      DUMPDS ALLOC=ACTIVE
  ```

Note: These steps can be performed after IPL using the DUMPDS command from an operator console, or early in IPL by putting the commands in the COMMNDxx parmlib member and pointing to the member from the IEASYSxx parmlib member using CMD=xx.

If you use COMMNDxx, you may want to specify DUMP=NO in the IEASYSxx parmlib member to prevent dumps taken during IPL from being written to SYS1.DUMPxx data sets.



Diagnostic Data - Traces

- GTF Trace
  - GTF cataloged procedure
  - GTFPARM parmlib member
  - Start and stop GTF trace
    - S GTF.EXAMPLE1 - P EXAMPLE1
- Other system traces
  - Component Trace
  - Master Trace
  - GFS Trace
  - System Trace

Figure 166. Diagnostic data - traces

8.7 Diagnostic data - traces

Another useful source of diagnostic data is the *trace*. Tracing collects information that identifies ongoing events that occur over a period of time. Some traces are running all the time so that trace data will be available in the event of a failure. Other traces must be explicitly started to trace a defined event. Some different trace types are shown on the visual.

8.7.1 GTF trace

Use a GTF trace to show system processing through events occurring in the system over time. The installation controls which events are traced. GTF tracing uses more resources and processor time than a system trace. Use GTF when you′re familiar enough with the problem to pinpoint the one or two events required to diagnose your system problem. GTF can be run to an external data set as well as a buffer.

When you activate GTF, it operates as a system task, in its own address space. The only way to activate GTF is to enter a START GTF command from a console with master authority. Using this command, the operator selects either IBM′s or your cataloged procedure for GTF. The cataloged procedure defines GTF operation; you can accept the defaults that the procedure establishes, or change the defaults by having the operator specify certain parameters on the START GTF command.

Because GTF sends messages to a console with master authority, enter the command only on a console that is eligible to be a console with master authority. Otherwise, you cannot view the messages from GTF that verify trace options and other operating information.

IBM supplies the GTF cataloged procedure, which resides in SYS1.PROCLIB. The cataloged procedure defines GTF operation, including storage needed, where output is to go, recovery for GTF, and the trace output data sets. Figure 167 shows the content of the IBM supplied cataloged procedure.

```
//GTF     PROC MEMBER=GTFPARM
//IEFPROC EXEC PGM=AHLGTF,PARM='MODE=EXT,DEBUG=NO,TIME=YES',
//  TIME=1440,REGION=2880K
//IEFRDER DD   DSNAME=SYS1.TRACE,UNIT=SYSDA,SPACE=(TRK,20),
//             DISP=(NEW,KEEP)
//SYSLIB  DD   DSN=SYS1.PARMLIB(&MEMBER),DISP=SHR
```

Figure 167. IBM-supplied GTF cataloged procedure

IBM supplies the GTFPARM parmlib member, which contains the GTF trace options shown below:

```
TRACE=SYSM,USR,TRC,DSP,PCI,SRM
```

These options request that GTF trace the following:

SYSM Selected system events

USR User data that the GTRACE macro passes to GTF

TRC Trace events associated with GTF itself

DSP Dispatchable units of work

PCI Program-controlled I/O interruptions

SRM Trace data associated with the system resource manager (RSM)

Note: Details regarding other GTF options can be found in *OS/390 MVS Diagnosis: Tools and Service Aids*, SY28-1085.

8.7.1.1 Starting GTF

To invoke GTF, the operator enters the START command shown below:

```
    {START|S}{GTF|membername}.identifier.
```

After the operator enters the START command, GTF issues message AHL100A or AHL125A to allow the operator either to specify or to change trace options. If the cataloged procedure or START command did not contain a member of predefined options, GTF issues message AHL100A so the operator may enter the trace options you want GTF to use. If the procedure or command did include a member of predefined options, GTF identifies those options by issuing the console messages AHL121I and AHL103I. Then you can either accept these options, or reject them and have the operator respecify new options. This sequence of messages appears as:

```
AHL121I TRACE OPTION INPUT INDICATED FROM MEMBER memname OF PDS dsname
AHL103I TRACE OPTIONS SELECTED -
keywd=(value),...,keywd=(value)
keywd,keywd,...,keywd
AHL125A RESPECIFY TRACE OPTIONS OR REPLY U
```

Alternatively, Figure 168 on page 278 shows the messages issued when a GTF procedure specifies the member that contains the parameters to be used.

```
START GTF.EXAMPLE1
AHL121I TRACE OPTION INPUT INDICATED FROM MEMBER GTFPARM OF PDS SYS1.PARMLIB
TRACE=SYSM,USR,TRC,DSP,PCI,SRM
AHL103I TRACE OPTIONS SELECTED--SYSM,USR,TRC,DSP,PCI,SRM
*451 AHL125A RESPECIFY TRACE OPTIONS OR REPLY U
REPLY 451,U
AHL031I GTF INITIALIZATION COMPLETE
```

Figure 168. Example: starting GTF with a cataloged procedure

The GTFPARM member contained:

```
TRACE=SYSM,USR,TRC,DSP,PCI,SRM
```

as shown in Figure 168.

8.7.1.2 Stopping GTF

The operator can enter the STOP command at any time during GTF processing. The amount of time you let GTF run depends on your installation and the problem you are trying to capture, but a common time is between 15 and 30 minutes.

To stop GTF processing, have the operator enter the STOP command. The STOP command must include either the GTF identifier specified in the START command, or the device number of the GTF trace data set if you specified MODE=EXT or MODE=DEFER to direct output to a data set.

If you are not sure of the identifier, or the device number of the trace data set, ask the operator to enter the following command:

```
DISPLAY A,LIST
```

In the following example of DISPLAY A,LIST output, the identifier for GTF is EVENT1.

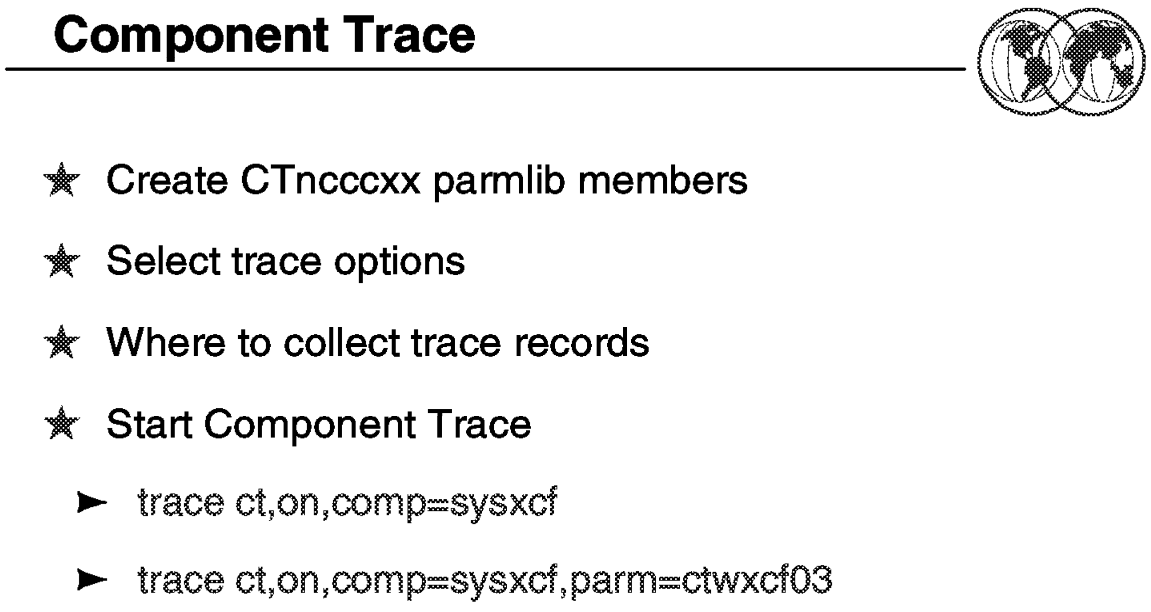
```
 IEE114I 14.51.49 1996.181 ACTIVITY     FRAME LAST   F      E   SYS=SY1
  JOBS     M/S    TS USERS    SYSAS    INITS   ACTIVE/MAX VTAM     OAS
 00000    00003    00000      00016    00000    00000/00000      00000
  LLA      LLA      LLA      NSW  S  VLF      VLF      VLF      NSW  S
  JES2     JES2     IEFPROC  NSW  S
  GTF      EVENT1   IEFPROC  NSW  S
```

Figure 169. Example: DISPLAY ACTIVE,LIST

The operator must enter the STOP command at a console with master authority. The general format of the STOP command is:

```
{STOP|P} identifier
```

When the STOP command takes effect, the system issues message AHL006I. If the system does not issue message AHL006I, then GTF tracing continues, remaining active until a STOP command takes effect or the next initial program load (IPL). When this happens, you will not be able to restart GTF tracing. In this case, you can use the FORCE ARM command to stop GTF. If there were several functions started with the same identifier on the START command, using the same identifier on the STOP command will stop all those functions.



Component Trace

- Create CTncccxx parmlib members
- Select trace options
- Where to collect trace records
- Start Component Trace
  - trace ct,on,comp=sysxcf
  - trace ct,on,comp=sysxcf,parm=ctwxcf03

Figure 170. Component trace

8.7.2 Component trace

The component trace service provides a way for MVS components to collect problem data about events. Each component that uses the component trace service has set up its trace in a way that provides the unique data needed for the component.

A component trace provides data about events that occur in the component. The trace data is intended for the IBM Support Center, which can use the trace to:

- Diagnose problems in the component
- See how the component is running

You will typically use component trace while re-creating a problem.

Component tracing shows processing within an MVS component. Typically, you might use component tracing while recreating a problem. The installation, with advice from the IBM Support Center, controls which events are traced for a system component. GTF does not have to be active to run a component trace.

8.7.2.1 Parmlib members

There is a table in *OS/390 MVS Diagnosis: Tools and Service Aids*, SY28-1085, that shows if a component has a parmlib member. It indicates if the member is a default member needed at system or component initialization, and if the component has default tracing. Some components run default tracing at all times when the component is running; default tracing is usually minimal and covers only unexpected events. Other components run traces only when requested. When preparing your production SYS1.PARMLIB system library, do the following:

- Make sure the parmlib contains all default members identified in the table. If parmlib does not contain the default members at initialization, the system issues messages.
- Make sure that the IBM-supplied CTIITT00 member is in the parmlib. PARM=CTIITT00 can be specified on a TRACE CT command for a component trace that does not have a parmlib member; CTIITT00 prevents the system from prompting for a REPLY after the TRACE CT command. In a sysplex, CTIITT00 is useful to prevent each system from requesting a reply.

An example for a parmlib definition for OS/390 UNIX is:

```
CTncccxx    -
CTIBPX00    -    OS/390 UNIX parmlib member
                 (which must be specified in BPXPRM00 member)
```

Where BPX is the ccc, and 00 is the xx and I is the n. For some components, you need to identify the component′s CTncccxx member in another parmlib member. See the parmlib member listed in the default member column in the table in *OS/390 MVS Diagnosis: Tools and Service Aids*, SY28-1085.

8.7.2.2 Trace options

If the IBM Support Center requests a trace, the Center might specify the options, if the component trace uses an OPTIONS parameter in its parmlib member or REPLY for the TRACE CT command.

You must specify all options you would like to have in effect when you start a trace. Options specified for a previous trace of the same component do not continue to be in effect when the trace is started again. If the component has default tracing started at initialization by a parmlib member without an OPTIONS parameter, you can return to the default by doing one of the following:

- Stopping the tracing with a TRACE CT,OFF command.
- Specifying OPTIONS() in the REPLY for the TRACE CT command or in the CTncccxx member.

8.7.2.3 Collecting trace records

Depending on the component, the potential locations of the trace data are:

- In address-space buffers, which are obtained in a dump
- In data-space buffers, which are obtained in a dump
- In a trace data set or sets, if supported by the component trace

If the trace records of the trace you want to run can be placed in more than one location, you need to select the location. For a component that supports trace data sets, you should choose trace data sets for the following reasons:

- Because you expect a large number of trace records
- To avoid interrupting processing with a dump of the trace data
- To keep the buffer size from limiting the amount of trace data
- To avoid increasing the buffer size

8.7.2.4 Starting component trace

Select how the operator is to request the trace. The component trace is started by either of the following:

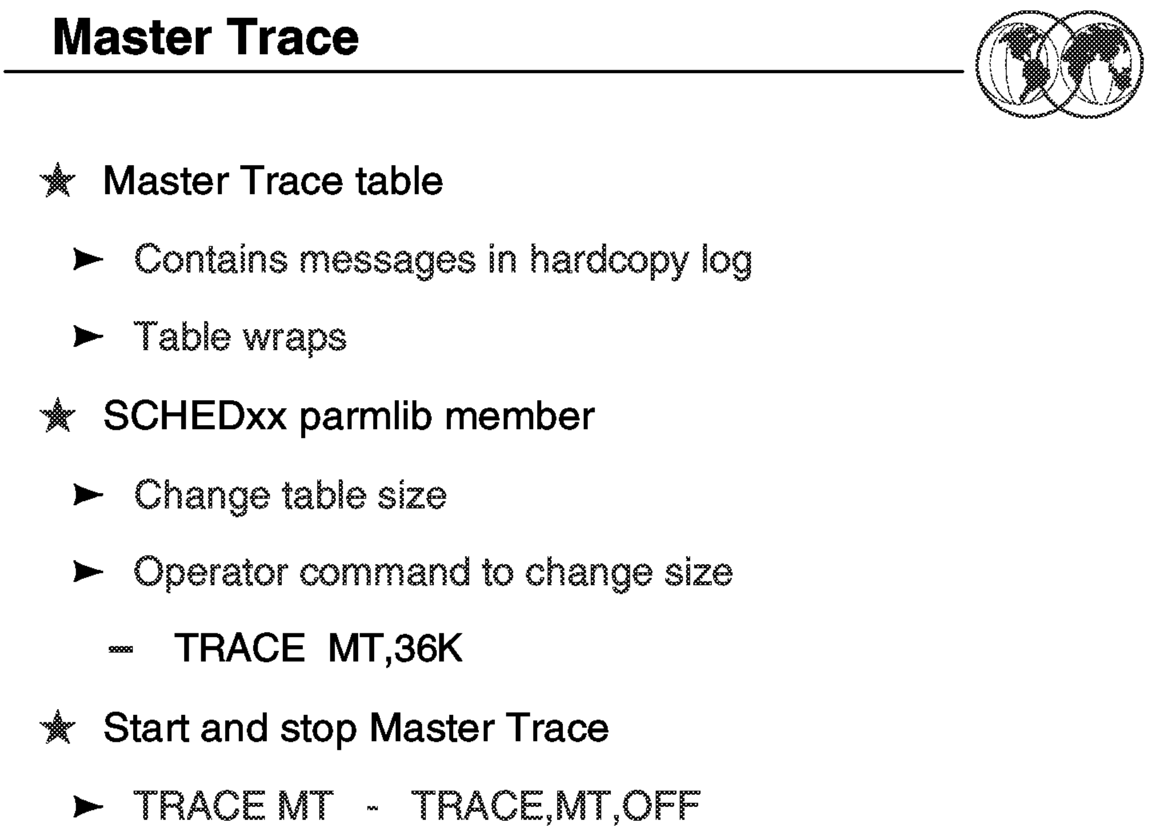
- A TRACE CT operator command without a PARM parameter, followed by a reply containing the options
- A TRACE CT operator command with a PARM parameter that specifies a CTncccxx parmlib member containing the options

To start a component trace, the operator enters a TRACE operator command on the console with MVS master authority. The operator replies with the options that you specified.

```
trace ct,on,comp=sysxcf
* 21 ITT006A ....
r 21,options=(serial,status),end
```

This example requests the same trace using parmlib member CTWXCF03. When TRACE CT specifies a parmlib member, the system does not issue message ITT006A.

```
trace ct,on,comp=sysxcf,parm=ctwxcf03
```



Master Trace

- Master Trace table
  - Contains messages in hardcopy log
  - Table wraps
- SCHEDxx parmlib member
  - Change table size
  - Operator command to change size
    - TRACE MT,36K
- Start and stop Master Trace
  - TRACE MT - TRACE,MT,OFF

Figure 171. Master trace

8.7.3 Master trace

Master trace maintains a table of the system messages that are routed to the hardcopy log. This creates a log of external system activity, while the other traces log internal system activity. Master trace is activated automatically at system initialization, but you can turn it on or off using the TRACE command.

Master trace can help you diagnose a problem by providing a log of the most recently issued system messages. For example, master trace output in a dump contains system messages that may be more pertinent to your problem than the usual component messages issued with a dump.

Use the master trace to show the messages to and from the master console. Master trace is useful because it provides a log of the most recently-issued messages. These can be more pertinent to your problem than the messages accompanying the dump itself. Master tracing is usually activated at IPL time and the data can be reviewed with IPCS and is saved when an SVC dump or stand-alone dump is taken.

At initialization, the master scheduler sets up a master trace table of 24 kilobytes. A 24-kilobyte table holds about 336 messages, assuming an average length of 40 characters. You can change the size of the master trace table or specify that no trace table be used by changing the parameters in the SCHEDxx member in SYS1.PARMLIB.

You can also change the size of the table using the TRACE command. For example, to change the trace table size to 36 kilobytes, enter:

```
TRACE MT,36K
```

Start, change, or stop master tracing by entering a TRACE operator command from a console with master authority. For example, to start the master tracing:

```
TRACE MT
```

To stop master tracing:

```
TRACE MT,OFF
```

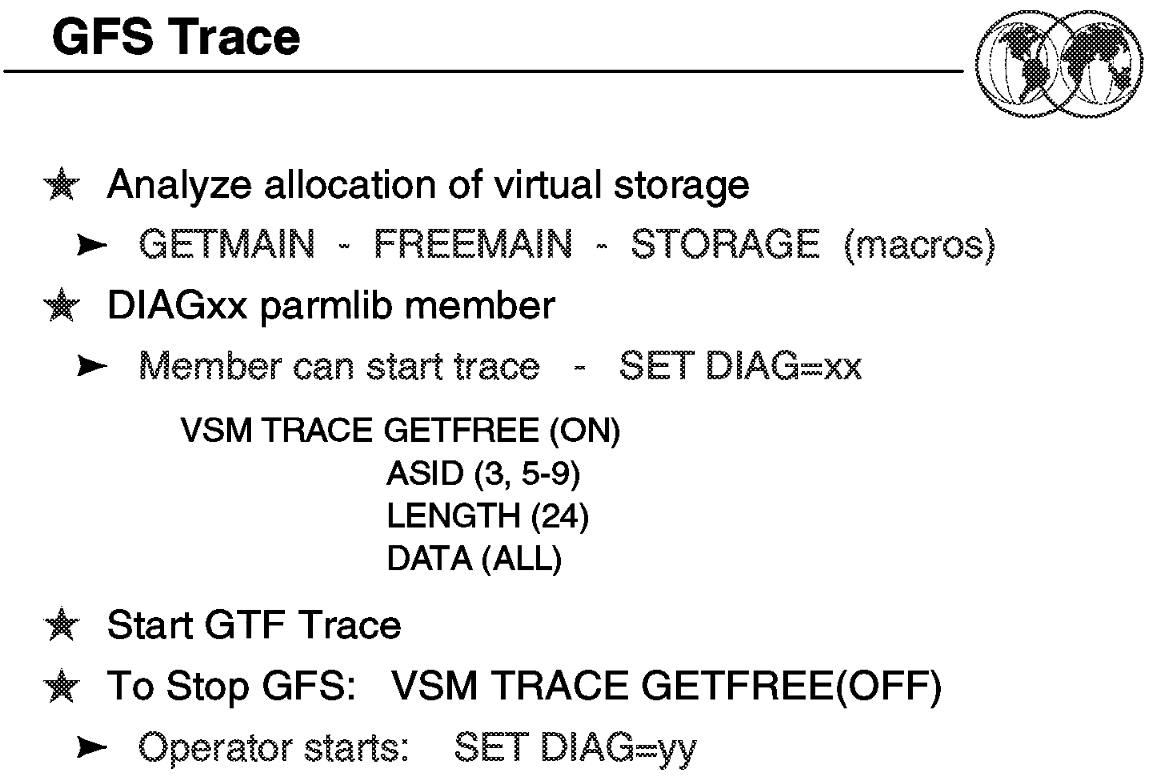
You can also use the TRACE command to obtain the current status of the master trace. The system displays the status in message IEE839I. For example, to ask for the status of the trace, enter:

```
TRACE STATUS
```

Figure 172 shows a sample of the master trace table. This is an in-storage copy of the system log (SYSLOG) and the amount of data contained in the table is dependant on the size of the table.

```
*** MASTER TRACE TABLE ***
TAG  IMM DATA  |------------------------ MESSAGE DATA ------------------
0001 008AD6A8  N 0000000 SP21     87153 17:59:11.88 STC  877 0000000
               IEF285I   VOL SER NOS= D83LOG.
0001 008AD6A8  N 0000000 SP21     87153 17:59:11.99          0000000
               IEA989I SLIP TRAP ID=X33E MATCHED
0001 008C7EF8  N 1000000 SP21     87153 17:59:12.22 JOB  801 0000000
               IEF234E D 2C3,338003,PUB
0001 008B51D8  N 1000000 SP21     87153 17:59:12.40 JOB  801 0000000
               IEF281I 2C3 NOW OFFLINE
0001 008B52FC  NR0000000 SP21     87153 17:59:12.41 MASTER   00000000
               IEE794I 2C6      PENDING OFFLINE
0001 008B52FC  N 1000000 SP21     87153 17:59:13.53 JOB  801 0000000
               IEF234E D 2C4,338004,STR
0001 008B51D8  N 1000000 SP21     87153 17:59:14.21 JOB  801 0000000
               IEF281I 2C4 NOW OFFLINE
0001 008C7DD4  M 0002000 SP21     87153 17:59:14.29 STC  828 0000000
               System  SYSIEFSD Q4<--------------------------------
0001 008B4D48  D                                         372 0000000
               Run  Exc  N1OFF2C0  SP21
0001 008B4DA8  D                                         372 0000000
               Wait Shr  *MASTER*  SP21        0:41:22
0001 008B39E4  D                                         372 0000000
               System  SYSIEFSD VARYDEV<---------------------------
```

Figure 172. Master trace formatted output



GFS Trace

- Analyze allocation of virtual storage
  - GETMAIN - FREEMAIN - STORAGE (macros)
- DIAGxx parmlib member
  - Member can start trace - SET DIAG=xx

```
VSM TRACE GETFREE (ON)
                  ASID (3, 5-9)
                  LENGTH (24)
                  DATA (ALL)
```

- Start GTF Trace
- To Stop GFS: VSM TRACE GETFREE(OFF)
  - Operator starts: SET DIAG=yy

Figure 173. GFS trace

8.7.4 GFS trace

Use GFS trace to collect information about requests for virtual storage via the GETMAIN, FREEMAIN, and STORAGE macro. GTF must be active to run a GFS trace. The following procedure explains how to request a GFS trace.

1. In the DIAGxx parmlib member, set the *VSM TRACE GETFREE* parameter to *ON* and define the GFS trace control data.

Example: DIAGxx parmlib member for starting GFS tracing

The following DIAGxx parmlib member starts GFS trace and limits the trace output to requests to obtain or release virtual storage that is 24 bytes long and resides in address spaces 3, 5, 6, 7, 8, and 9:

```
VSM TRACE GETFREE (ON)
        ASID (3, 5-9)
        LENGTH (24)
        DATA (ALL)
```

Note: If you want the IPCS GTFTRACE output to be formatted, you must include the TYPE and FLAGS data items on the DATA keyword specification of the DIAGxx parmlib member.

Note: You′ll need another DIAGxx parmlib member defined to stop GFS tracing specifying:

```
VSM TRACE GETFREE (OFF)
```

2. Ask the operator to enter the SET DIAG=xx command to activate GFS trace using the definitions in the DIAGxx parmlib member.

3. Start a GTF trace (ask the operator to enter a START membername command on the master console). The membername is the name of the member that contains the source JCL (either a cataloged procedure or a job). Tell the operator to specify a user event identifier X′F65′ to trace GTF user trace records.

Example: Starting a GTF trace for GFS data

In the following example, the operator starts GTF tracing with cataloged procedure GTFPROC to get GFS data in the GTF trace output. The contents of cataloged procedure GTFPROC are as follows:

```
//GTF     PROC MEMBER=GTFPROC
//* Starts GTF
//IEFPROC EXEC PGM=AHLGTF,REGION=32M,TIME=YES,
//  PARM=′MODE=EXT,DEBUG=NO,TIME=YES,BLOK=40K,SD=0K,SA=40K′
//IEFRDER DD DSN=MY.GTF.TRACE,
//           DISP=SHR,UNIT=3390,VOL=SER=VOL001
```

The operator then replies to messages AHL100A with the USRP option. When message AHL101A prompts the operator for the keywords for option USRP, the operator replies with USR=(F65) to get the GFS user trace records in the GTF trace output.

4. To stop the GTF trace, ask the operator to enter a STOP procname command on the master console.

5. To stop GFS trace, create a DIAGxx parmlib member with:

   VSM TRACE GETFREE(OFF) and have the operator enter a
   SET DIAG=xx command.

GTF places the GFS trace data in a user trace record with event identifier X′F65′. To obtain GFS trace data, do one of the following:

1. When GTF writes the trace data to a data set, format and print the trace data with the IPCS GTFTRACE subcommand.

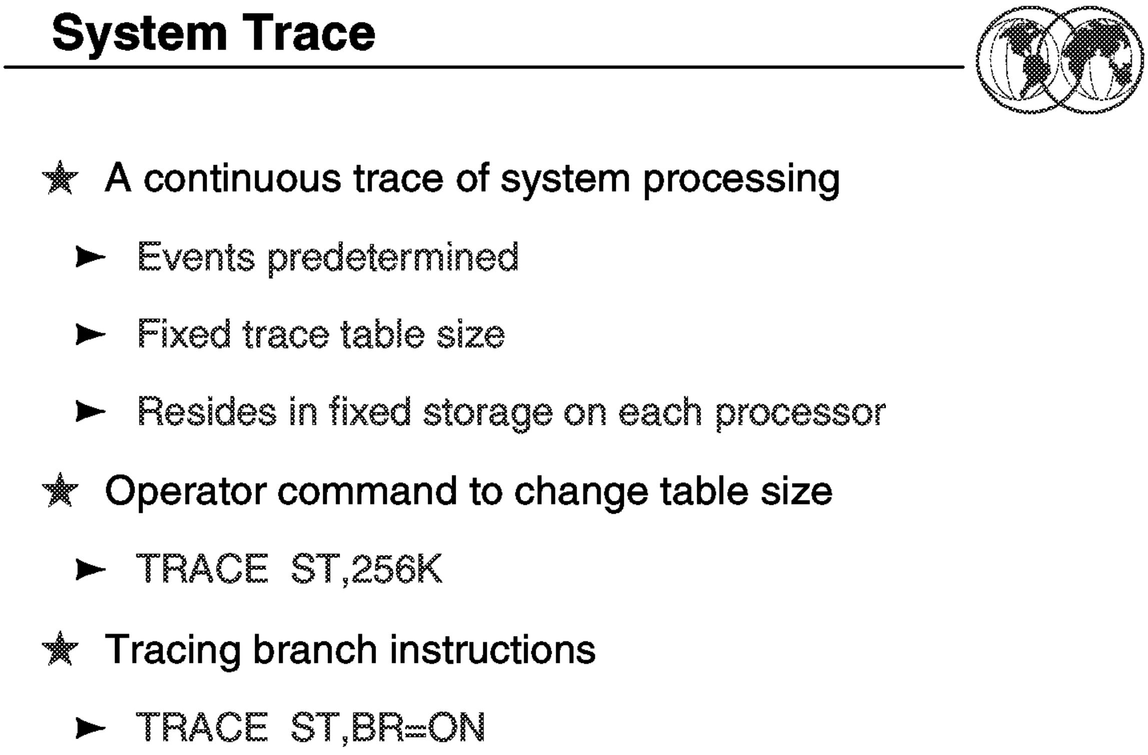
2. When GTF writes trace data only in the GTF address space, use IPCS to see the data in an SVC dump. Request the GTF trace data in the dump through the SDATA=TRT dump option.

3. Issue the IPCS GTFTRACE subcommand to format and see the trace in an unformatted dump.

Figure 174 on page 287 shows an example of formatted GFS trace data.

```
IPCS
   GTFTRACE DA('MY.GTF.TRACE') USR(F65)
IKJ56650I TIME-03:42:20 PM. CPU-00:00:01 SERVICE-52291 SESSION-00:00:20
BLS18122I Initialization in progress for DSNAME('MY.GTF.TRACE')
IKJ56650I TIME-03:42:21 PM. CPU-00:00:01 SERVICE-54062 SESSION-00:00:20
   **** GTFTRACE DISPLAY OPTIONS IN EFFECT ****
 USR=SEL

**** GTF DATA COLLECTION OPTIONS IN EFFECT: ****
 USRP option

                 **** GTF TRACING ENVIRONMENT ****
       Release: SP6.0.6   FMID: HBB6606   System name: CMN
       CPU Model: 9672  Version: FF  Serial no. 270067


 USRDA F65 ASCB 00FA0800              JOBN MYGTF2
 Getmain SVC(120)  Cond=Yes
 Loc=(Below,Below)  Bndry=Dblwd
 Return address=849CA064  Asid=001A  Jobname=MYGTF2
 Subpool=229  Key=0  Asid=001A  Jobname=MYGTF2    TCB=008DCA70 Retcode=0
 Storage address=008D6768 Length=10392 X'2898'
   GPR Values
       0-3  00002898  00000000  7FFFC918  0B601E88
       4-7  01FE3240  008FF830  849CA000  00FA0800
       8-11 00000000  00000DE8  049CBFFE  849CA000
      12-15 049CAFFF  0B601A9C  00FE9500  0000E510
               GMT-01/06/1998 21:15:43.111628  LOC-01/06/1998 21:15:43.1

USRDA F65 ASCB 00FA0800              JOBN MYGTF2
Freemain SVC(120)  Cond=No
Return address=8B2D608A  Asid=001A  Jobname=MYGTF2
Subpool=230  Key=0  Asid=001A  Jobname=MYGTF2    TCB=008DCA70  Retcode=0
Storage address=7F73DFF8  Length=8 X'8'
  GPR Values
      0-3  00000000  7F73DFF8  008D82D8  008D7BC0
      4-7  008D8958  008D6B08  008D85C8  0B335000
      8-11 00000002  00000000  7F73DFF8  008D862C
     12-15 8B2D6044  008D8C98  849D242A  0000E603

               GMT-01/06/1998 21:15:43.111984  LOC-01/06/1998 21:15:43.1
```

Figure 174. Example of formatted GFS trace output



System Trace

- A continuous trace of system processing
  - Events predetermined
  - Fixed trace table size
  - Resides in fixed storage on each processor
- Operator command to change table size
  - TRACE ST,256K
- Tracing branch instructions
  - TRACE ST,BR=ON

Figure 175. System trace

8.7.5 System trace

Use system trace to see system processing through events occurring in the system over time. System tracing is activated at initialization and, typically, runs continuously. It records many system events, with minimal detail about each. The events traced are predetermined, except for branch tracing. This trace uses fewer resources and is faster than a GTF trace.

System trace tables reside in fixed storage on each processor. The default trace table size is 64 kilobytes per processor, but you can change it using the TRACE ST command. We do not recommend running with trace tables smaller than the default 64 kilobytes. You might, however, want to increase the size of the system trace table from the default 64 kilobyte when:

- You find that the system trace does not contain tracing from a long enough time period.
- You want to trace branch instructions (using the BR=ON option on the TRACE ST command when you start tracing.)

Because system trace usually runs all the time, it is very useful for problem determination. While system trace and the general trace facility (GTF) lists many of the same system events, system trace also lists events occurring during system initialization, before GTF tracing can be started. System trace also traces branches and cross-memory instructions, which GTF cannot do.

Issue the following command to increase the system trace table size to 256K:

```
TRACE ST,256K
```

System tracing allows you the option of tracing branch instructions, such as BALR, BASR, BASSM, and BAKR, along with other system events.

Note: With branch tracing on can affect your system performance and use very large amounts of storage. Do not use branch tracing as the default for system tracing on your system. You should only use it for short periods of time to solve a specific problem. The default system tracing does not include branch instructions.

When you want to trace branch instructions, do the following:

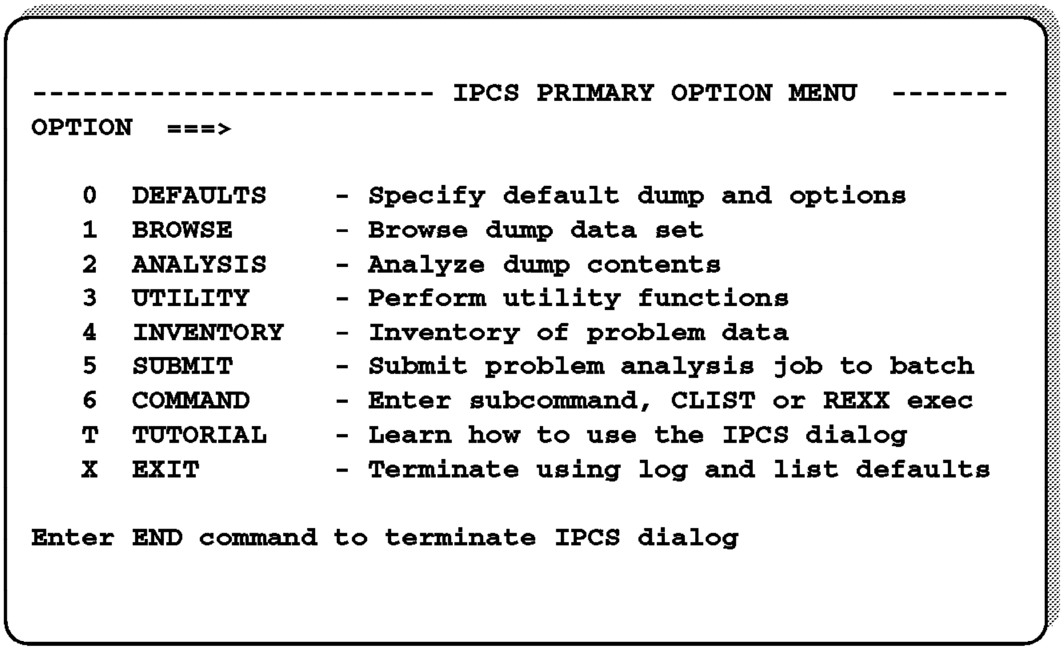
```
TRACE ST,BR=ON
```

System tracing will be captured in all dump situations by default, except during a SNAP dump where SDATA=TRT must be specified.

```
------------------------SYSTEM TRACE TABLE ---------------------------------
PR ASID  WU-ADDR  IDENT CD/D PSW----- ADDRESS---UNIQUE-1 UNIQUE-2 UNIQUE-3
                                                UNIQUE-4 UNIQUE-5 UNIQUE-6

01 000C 00AFF090   PGM   011 071C2000 81C80EB2  00020011 00ACFA18
01 000C 00AFF090   CLKC      071C2000 81C825A8  00001004 00000000
01 000C 00AFF090   DSP       071C2000 81C825A8  00000000 00000F80
01 000C 00AFF090   SVC    78 071C1000 81C56242  0000ED12 000000C0 00000000
01 000C 00AFF090   SSCH  271 00  02   00F3E12C  00F5FDF8 00008000
01 000C 00AFF090   SVCR    0 070C2000 00EB19C2  00000000 00F1845F 00000000
01 000C 00AFF090   SVC     1 070C2000 00EB19CC  00000000 00000001
01 000C 00AFF090   SVCR    1 070C2000 00EB19CC  80AEAF20 00000001
01 0001 00000000   I/O   271 070E0000 00000000  00004007 00CEFE58
```

Figure 176. SYSTRACE output example

Interactive Problem Control System (IPCS)



```
----------------------- IPCS PRIMARY OPTION MENU  -------
OPTION  ===>

   0  DEFAULTS    - Specify default dump and options
   1  BROWSE      - Browse dump data set
   2  ANALYSIS    - Analyze dump contents
   3  UTILITY     - Perform utility functions
   4  INVENTORY   - Inventory of problem data
   5  SUBMIT      - Submit problem analysis job to batch
   6  COMMAND     - Enter subcommand, CLIST or REXX exec
   T  TUTORIAL    - Learn how to use the IPCS dialog
   X  EXIT        - Terminate using log and list defaults

Enter END command to terminate IPCS dialog
```

Figure 177. Interactive Problem Control System (IPCS)

8.8 Interactive Problem Control System (IPCS)

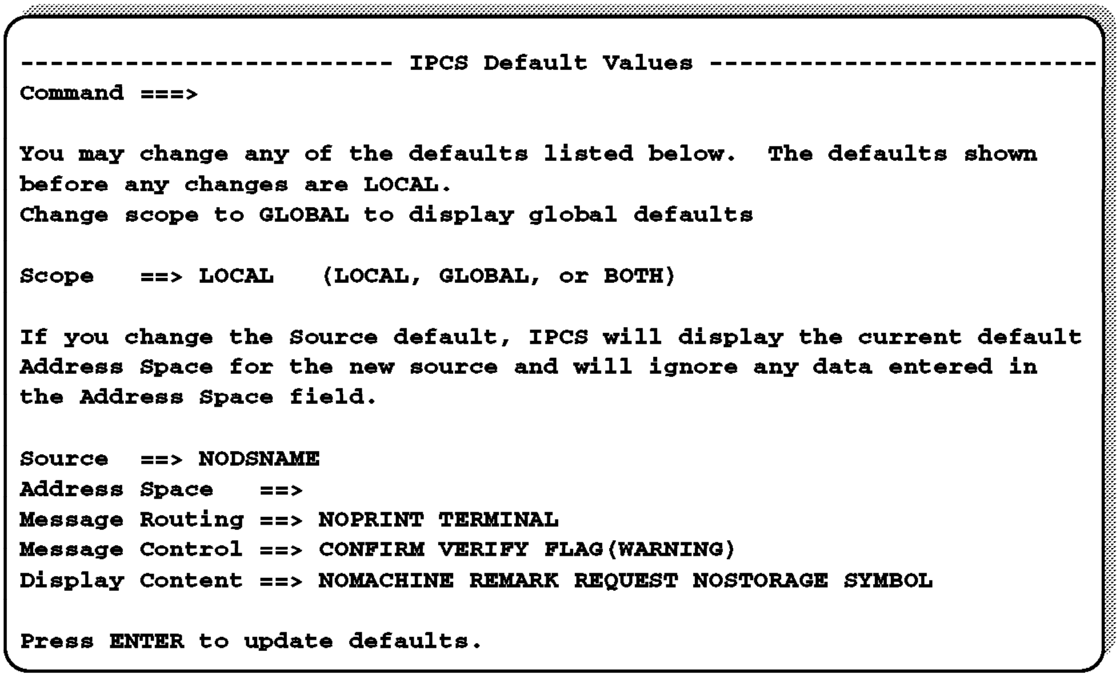
The most powerful diagnostic tool at your disposal is Interactive Program Control System (IPCS). IPCS is a tool provided in the MVS system to aid in diagnosing software failures. IPCS provides formatting and analysis support for dumps and traces produced by MVS, other program products, and applications that run on MVS.

SVC dumps, stand-alone dumps, and some traces are unformatted and need to be formatted before any analysis can begin. IPCS provides the tools to format dumps and traces in both an online and batch environment. IPCS provides you with commands that will let you interrogate specific components of the operating system and allows you to review storage locations associated with an individual task or control block. IPCS allows you to quickly review and isolate key information that will assist with your problem determination process.

IPCS can be invoked from ISPF running under TSO. You can create an option on an ISPF selection panel for starting the IPCS dialog. The to start the IPCS dialog, select the appropriate option.

The visual shows the IPCS primary option menu.

Setting the IPCS Defaults



```
------------------------ IPCS Default Values -------------------------
Command ===>

You may change any of the defaults listed below.  The defaults shown
before any changes are LOCAL.
Change scope to GLOBAL to display global defaults

Scope   ==> LOCAL   (LOCAL, GLOBAL, or BOTH)

If you change the Source default, IPCS will display the current default
Address Space for the new source and will ignore any data entered in
the Address Space field.

Source  ==> NODSNAME
Address Space   ==>
Message Routing ==> NOPRINT TERMINAL
Message Control ==> CONFIRM VERIFY FLAG(WARNING)
Display Content ==> NOMACHINE REMARK REQUEST NOSTORAGE SYMBOL

Press ENTER to update defaults.
```

Figure 178. Setting the IPCS defaults

8.8.1 Setting the IPCS defaults

Option **0** from the Primary Option Menu enables you to identify the data set that contains the dump you will be analyzing. The visual shows the IPCS default option menu.

Change the **SOURCE ==> NODSNAME** parameter to identify the dump data set.

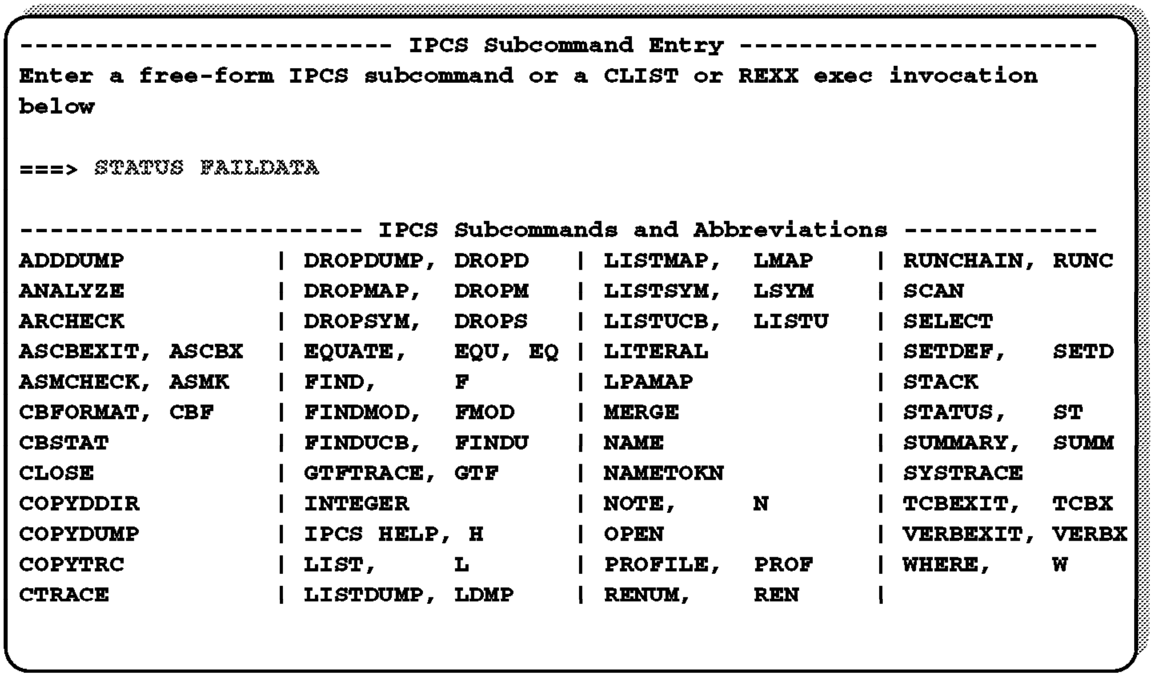
For example, **SOURCE ==> DSNAME(**′*xxxx.xxxxxx*′) will set the source DSNAME, where *xxxx.xxxxxx*, is the data set containing the dump to be reviewed.

When you press Enter, the following messages are issued during the formatting process:

```
TIME-05:14:53 AM. CPU-00:00:46 SERVICE-673781 SESSION-00:48:42 APRIL 13
Initialization in progress for DSNAME(′SYS1.DUMP03′)
TITLE=COMPID=DF115,CSECT=IGWLGMOT+1264,DATE=02/18/94,MAINTID= NONE
RC=00000024,RSN=12088C01
May summary dump data be used by dump access?  Enter Y to use, N to
bypass
Y           Note. Enter Yes
4,616 blocks, 19,202,560 bytes, in DSNAME(′SYS1.DUMP03′)
TIME-05:15:05 AM. CPU-00:00:46 SERVICE-702725 SESSION-00:48:53 APRIL 13
***
```

Figure 179. IPCS Dump initialisation messages

IPCS Subcommand Entry Panel



```
------------------------ IPCS Subcommand Entry -----------------------
Enter a free-form IPCS subcommand or a CLIST or REXX exec invocation
below

===> STATUS FAILDATA

---------------------- IPCS Subcommands and Abbreviations ------------
ADDDUMP           | DROPDUMP, DROPD   | LISTMAP,  LMAP    | RUNCHAIN, RUNC
ANALYZE           | DROPMAP,  DROPM   | LISTSYM,  LSYM    | SCAN
ARCHECK           | DROPSYM,  DROPS   | LISTUCB,  LISTU   | SELECT
ASCBEXIT, ASCBX   | EQUATE,   EQU, EQ | LITERAL           | SETDEF,   SETD
ASMCHECK, ASMK    | FIND,     F       | LPAMAP            | STACK
CBFORMAT, CBF     | FINDMOD,  FMOD    | MERGE             | STATUS,   ST
CBSTAT            | FINDUCB,  FINDU   | NAME              | SUMMARY,  SUMM
CLOSE             | GTFTRACE, GTF     | NAMETOKN          | SYSTRACE
COPYDDIR          | INTEGER           | NOTE,     N       | TCBEXIT,  TCBX
COPYDUMP          | IPCS HELP, H      | OPEN              | VERBEXIT, VERBX
COPYTRC           | LIST,     L       | PROFILE,  PROF    | WHERE,    W
CTRACE            | LISTDUMP, LDMP    | RENUM,    REN     |
```

Figure 180. Select the IPCS Subcommand Entry panel

8.8.2 Select the IPCS Subcommand Entry panel

Return to the IPCS Primary Option menu and select option **6**. When you press Enter, the visual shows the IPCS Subcommand Entry panel that is displayed.

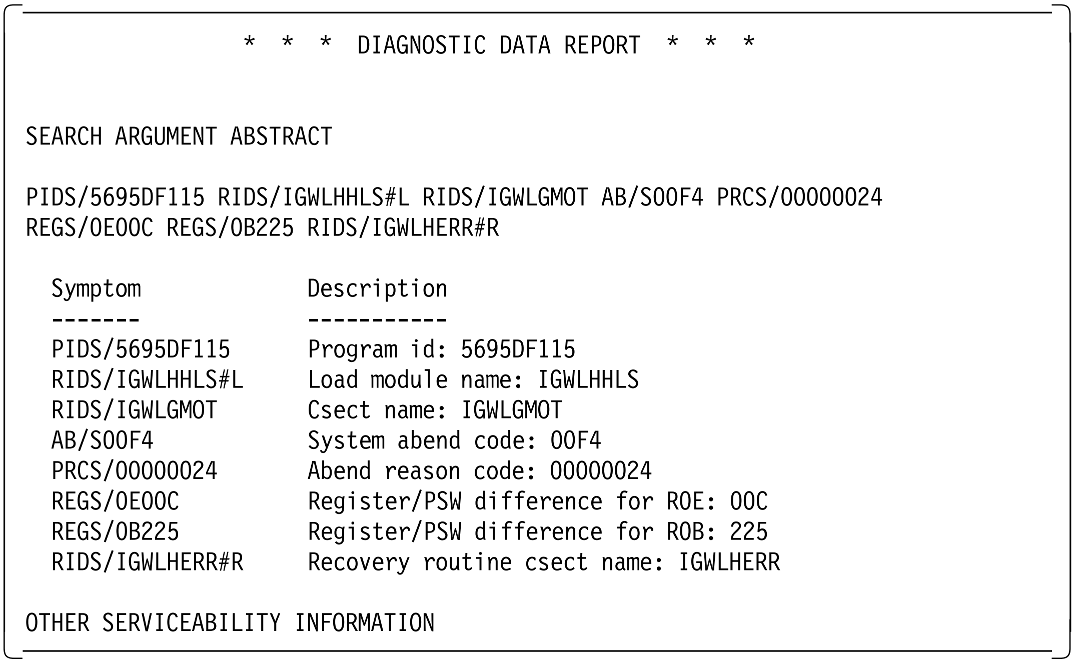
Use the IPCS subcommand STATUS FAILDATA, shown in the visual, to locate the specific instruction that failed and to format all the data in an SVC dump related to the software failure. This report gives information about the CSECT involved in the failure, the component identifier, and the PSW address at the time of the error.

Note: For SLIP dumps or CONSOLE dumps, use SUMMARY FORMAT or VERBEXIT LOGDATA instead of STATUS FAILDATA. Any valid IPCS command would have started the initialization process and the related display that results after initialization. It should be noted that the dump is only initialized the first time it is referenced via IPCS, and will only be initialized again if the dump is deleted from the IPCS inventory.

After the initialization process, the address space field in the IPCS Default Values panel will now contain the address space identifier (ASID) information stored in the dump data set SYS1.DUMP00. For example:

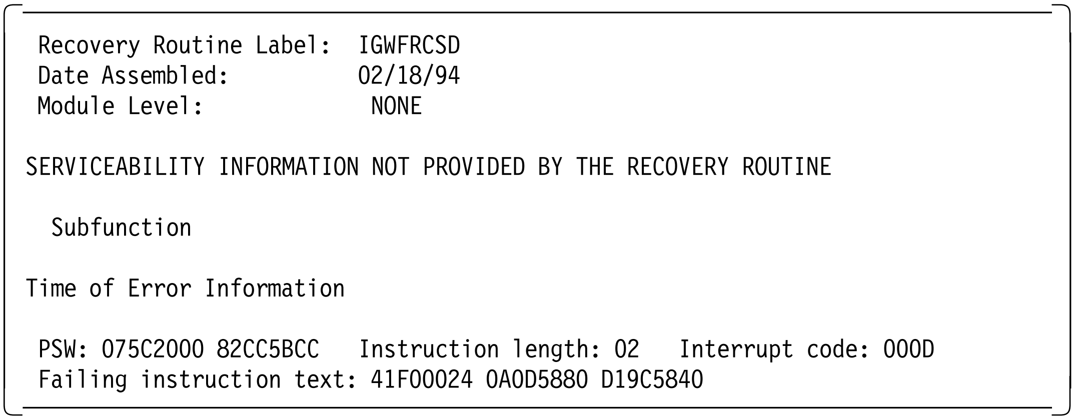
```
Address Space ==> ASID(X'009E')
```

The IPCS STATUS FAILDATA function shows us a diagnostic report that summarizes the failure. Figure 181 on page 293 and Figure 182 on page 293 show the FAILDATA information.



```
                * * *  DIAGNOSTIC DATA REPORT  * * *


SEARCH ARGUMENT ABSTRACT

PIDS/5695DF115 RIDS/IGWLHHLS#L RIDS/IGWLGMOT AB/S00F4 PRCS/00000024
REGS/0E00C REGS/0B225 RIDS/IGWLHERR#R

  Symptom             Description
  -------             -----------
  PIDS/5695DF115      Program id: 5695DF115
  RIDS/IGWLHHLS#L     Load module name: IGWLHHLS
  RIDS/IGWLGMOT       Csect name: IGWLGMOT
  AB/S00F4            System abend code: 00F4
  PRCS/00000024       Abend reason code: 00000024
  REGS/0E00C          Register/PSW difference for R0E: 00C
  REGS/0B225          Register/PSW difference for R0B: 225
  RIDS/IGWLHERR#R     Recovery routine csect name: IGWLHERR

OTHER SERVICEABILITY INFORMATION
```

Figure 181. STATUS FAILDATA display - screen 1



```
 Recovery Routine Label:  IGWFRCSD
 Date Assembled:          02/18/94
 Module Level:             NONE

SERVICEABILITY INFORMATION NOT PROVIDED BY THE RECOVERY ROUTINE

  Subfunction

Time of Error Information

 PSW: 075C2000 82CC5BCC   Instruction length: 02   Interrupt code: 000D
 Failing instruction text: 41F00024 0A0D5880 D19C5840
```

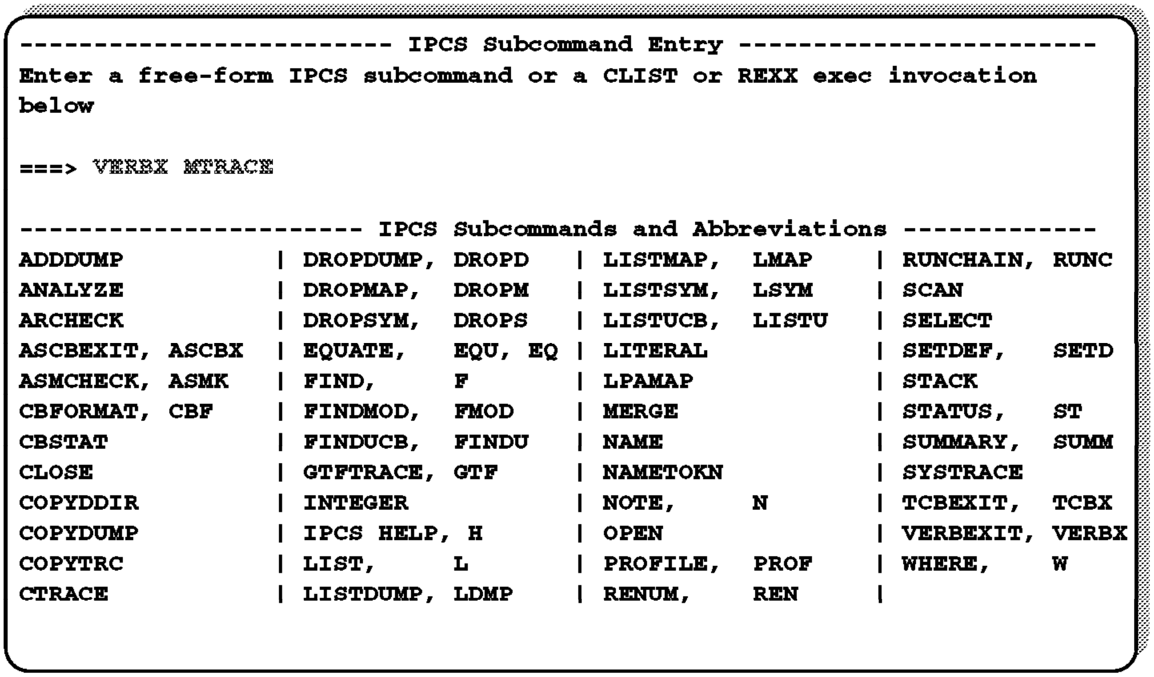
Figure 182. STATUS FAILDATA display - screen 2

Note: All the information is not included for screen 1 or screen 2; only the important information for this problem is shown. We now know that the load module that was pointed to by the program status word (PSW) was IGWLHHLS, the CSECT within that load module was IGWLGMOT, the abend code (0F4), and the abend reason code (0024).

Also, this information was also displayed during the initialization of the dump data set and, is shown in Figure 179 on page 291.

With the information we currently have we could perform a search of the IBM problem databases for a possible solution, but in this instance we will pursue the problem using IPCS to enable you to develop a better understanding of problem analysis techniques.

IPCS VERBX MTRACE Command



```
------------------------- IPCS Subcommand Entry ------------------------
Enter a free-form IPCS subcommand or a CLIST or REXX exec invocation
below

===> VERBX MTRACE

----------------------- IPCS Subcommands and Abbreviations -------------
ADDDUMP            | DROPDUMP, DROPD    | LISTMAP,  LMAP     | RUNCHAIN, RUNC
ANALYZE            | DROPMAP,  DROPM    | LISTSYM,  LSYM     | SCAN
ARCHECK            | DROPSYM,  DROPS    | LISTUCB,  LISTU    | SELECT
ASCBEXIT, ASCBX    | EQUATE,   EQU, EQ  | LITERAL            | SETDEF,   SETD
ASMCHECK, ASMK     | FIND,     F        | LPAMAP             | STACK
CBFORMAT, CBF      | FINDMOD,  FMOD     | MERGE              | STATUS,   ST
CBSTAT             | FINDUCB,  FINDU    | NAME               | SUMMARY,  SUMM
CLOSE              | GTFTRACE, GTF      | NAMETOKN           | SYSTRACE
COPYDDIR           | INTEGER            | NOTE,     N        | TCBEXIT,  TCBX
COPYDUMP           | IPCS HELP, H       | OPEN               | VERBEXIT, VERBX
COPYTRC            | LIST,     L        | PROFILE,  PROF     | WHERE,    W
CTRACE             | LISTDUMP, LDMP     | RENUM,    REN      |
```

Figure 183. IPCS VERBX MTRACE command

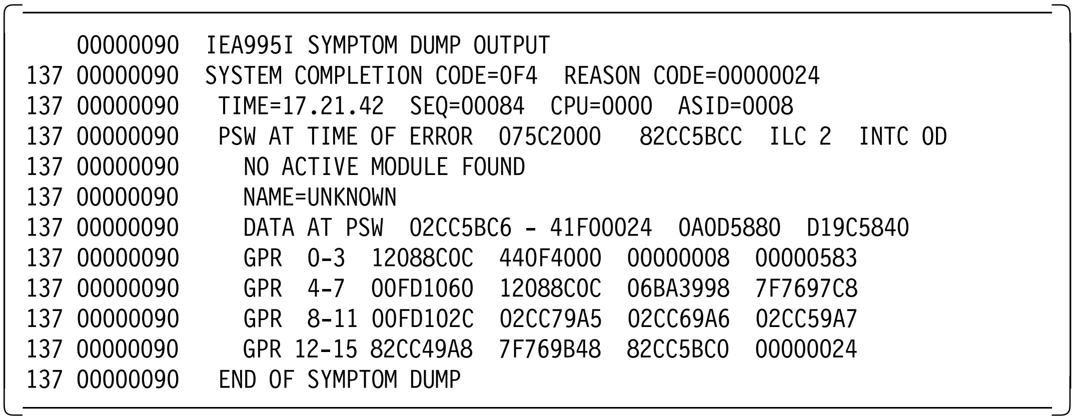
8.8.3 IPCS VERBX MTRACE command

The next command we will issue will be the first of our trace commands. Figure 184 on page 295 shows the VERBX MTRACE output. This is similar to the SYSLOG output.

The command displays the following:

- The master trace table entries for the dumped system. This table is a wraparound data area that holds the most recently issued console messages in a first-in, first-out order.

  Figure 184 on page 295 shows a small sample of what is contained in the MTRACE. In this sample we see details of the symptom dump for our problem.



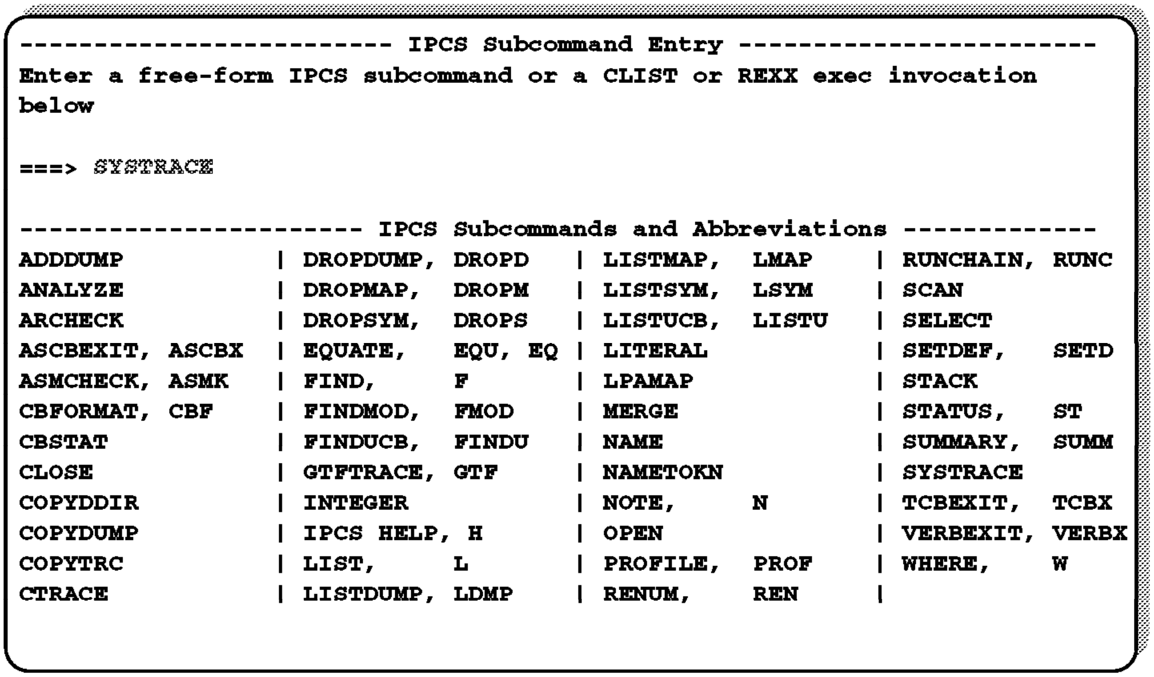
```
    00000090  IEA995I SYMPTOM DUMP OUTPUT
137 00000090  SYSTEM COMPLETION CODE=0F4  REASON CODE=00000024
137 00000090   TIME=17.21.42  SEQ=00084  CPU=0000  ASID=0008
137 00000090   PSW AT TIME OF ERROR  075C2000   82CC5BCC  ILC 2  INTC OD
137 00000090     NO ACTIVE MODULE FOUND
137 00000090     NAME=UNKNOWN
137 00000090     DATA AT PSW  02CC5BC6 - 41F00024  0A0D5880  D19C5840
137 00000090     GPR  0-3  12088C0C  440F4000  00000008  00000583
137 00000090     GPR  4-7  00FD1060  12088C0C  06BA3998  7F7697C8
137 00000090     GPR  8-11 00FD102C  02CC79A5  02CC69A6  02CC59A7
137 00000090     GPR 12-15 82CC49A8  7F769B48  82CC5BC0  00000024
137 00000090   END OF SYMPTOM DUMP
```

Figure 184. VERBX MTRACE output

All data that is displayed on the MVS master console will be captured in the master trace table. The amount of data kept is related to the master trace table buffer size, such as:

- The NIP hard-copy message buffer
- The branch entry and NIP time messages on the delayed issue queue

IPCS SYSTRACE Command



```
------------------------ IPCS Subcommand Entry -----------------------
Enter a free-form IPCS subcommand or a CLIST or REXX exec invocation
below

===> SYSTRACE

---------------------- IPCS Subcommands and Abbreviations ------------
ADDDUMP           | DROPDUMP, DROPD   | LISTMAP,  LMAP    | RUNCHAIN, RUNC
ANALYZE           | DROPMAP,  DROPM   | LISTSYM,  LSYM    | SCAN
ARCHECK           | DROPSYM,  DROPS   | LISTUCB,  LISTU   | SELECT
ASCBEXIT, ASCBX   | EQUATE,   EQU, EQ | LITERAL           | SETDEF,   SETD
ASMCHECK, ASMK    | FIND,     F       | LPAMAP            | STACK
CBFORMAT, CBF     | FINDMOD,  FMOD    | MERGE             | STATUS,   ST
CBSTAT            | FINDUCB,  FINDU   | NAME              | SUMMARY,  SUMM
CLOSE             | GTFTRACE, GTF     | NAMETOKN          | SYSTRACE
COPYDDIR          | INTEGER           | NOTE,     N       | TCBEXIT,  TCBX
COPYDUMP          | IPCS HELP, H      | OPEN              | VERBEXIT, VERBX
COPYTRC           | LIST,     L       | PROFILE,  PROF    | WHERE,    W
CTRACE            | LISTDUMP, LDMP    | RENUM,    REN     |
```

Figure 185. IPCS SYSTRACE command

8.8.4 IPCS SYSTRACE command

The system trace can be examined by issuing the SYSTRACE command from the IPCS subcommand entry panel shown in the visual. Issuing the SYSTRACE command on its own will display trace entries associated with the dumped ASID only. Issuing the SYSTRACE ALL command will display all system trace entries. To display the time field in local time, add the TIME(LOCAL) parameter. A complete system trace command is as follows:

```
SYSTRACE ALL TIME(LOCAL)
```

Figure 186 on page 297 shows a small sample of the system trace. The time stamps would appear on the right-hand side of the display.

```
                   SYSTRACE Example 1 (*SVC)
-------------------------------------------------------------------
CP ASID   TCB      TRACE ID         PSW           R15       R0   R1
--|----|--------|-----------|-----------------|---------|--------|-
00 0008 007FD720  *SVC     D 075C2000 82CC5BCC  00000024 12088C0C
00 0008 007FD720   SSRV   78          828BC3F0  0000FF50 000000C8
                                                00080000
00 0008 007FD720   SSRV   78          828BC41A  0000FF70 00000FB0
                                                00080000
00 0008 007FD720   EXT  1005 070C0000 813B54AC  00001005
-------------------------------------------------------------------
                   SYSTRACE Example 2 (*RCVY)
-------------------------------------------------------------------
00 0153 008DA530   SSRV   78          40E5269C  4050E612 000002B8
                                                01530000
00 0153 008DA530   SSRV   78          80E52704  4050E612 00000080
                                                01530000
02 0013 008C5E88  *RCVY PROG                    940C4000 00000011

02 0013 008C5E88   SSRV   78          8109CADC  4000EF50 00000818
                                                00010000
02 0013 008C5E88  *RCVY FRR  070C0000 9056FBE8  940C4000 00000011

02 0013 008C5E88   CLKC      070C0000 9056FBE8  00001004 00000000
```

Figure 186. System trace output

The SYSTRACE data can be best reviewed by going to the end of the trace output, and issuing a FIND "*SVC" PREV command. This should help you locate the trace entry that indicates the abend. Another useful trace point to search for is **RCVY* which indicates a recovery action. Entries prior to this can assist with problem diagnosis. An SVC D is the abend SVC. Note the PSW, which is the same as identified in previous steps. This will point to the next instruction to be processed.

The SVC trace entries are as follows:

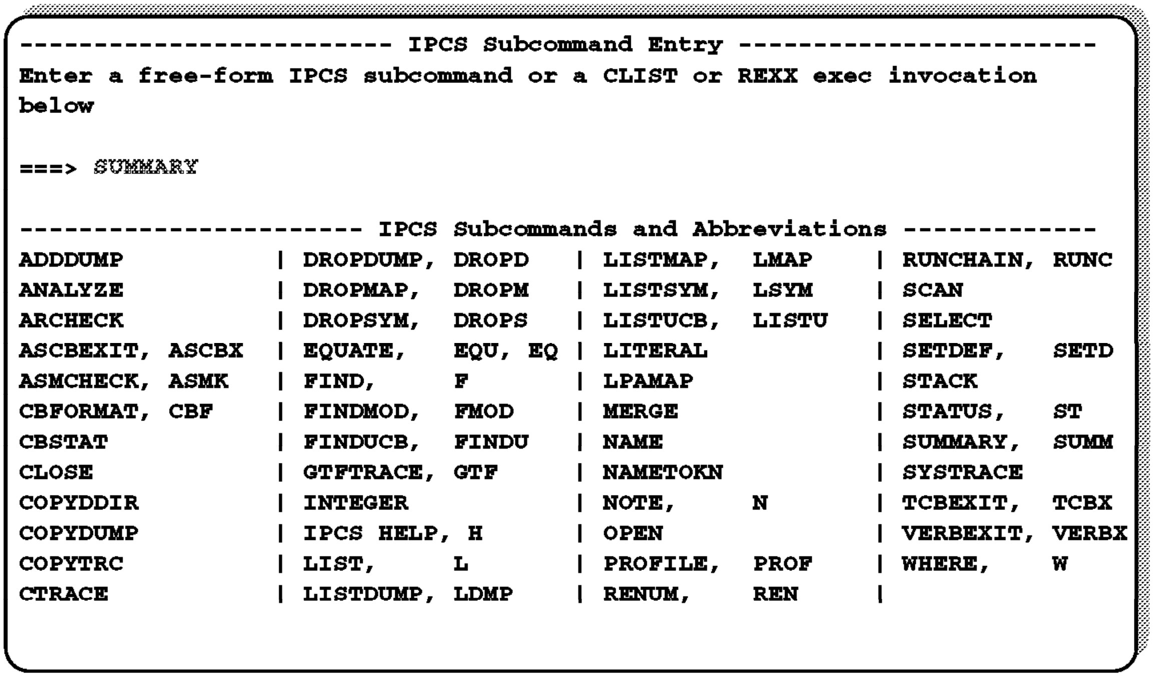
- An SVC trace entry is for processing of a Supervisor Call (SVC) instruction
- An SVCE trace entry is for an error during processing of an SVC instruction
- An SVCR trace entry is for return from SVC instruction processing

The actual SVC identified in the SYSTRACE is the hexadecimal identification. This must be converted to decimal to enable the correct research, for example:

- The SYSTRACE entry for SVC 78, would convert to a decimal SVC number of 120, which, when referencing *OS/390 MVS Diagnosis Reference*, SY28-1084, would identify the GETMAIN/FREEMAIN SVC.

This is an example of just one of the many trace entries that are created during the life of an OS/390 task. For a further explanation of other trace entries, you can reference *OS/390 Diagnosis: Tools and Service Aids*, SY28-1085.

IPCS SUMMARY Command



```
------------------------- IPCS Subcommand Entry ------------------------
Enter a free-form IPCS subcommand or a CLIST or REXX exec invocation
below

===> SUMMARY

---------------------- IPCS Subcommands and Abbreviations -------------
ADDDUMP           | DROPDUMP, DROPD   | LISTMAP,  LMAP    | RUNCHAIN, RUNC
ANALYZE           | DROPMAP,  DROPM   | LISTSYM,  LSYM    | SCAN
ARCHECK           | DROPSYM,  DROPS   | LISTUCB,  LISTU   | SELECT
ASCBEXIT, ASCBX   | EQUATE,   EQU, EQ | LITERAL           | SETDEF,   SETD
ASMCHECK, ASMK    | FIND,     F       | LPAMAP            | STACK
CBFORMAT, CBF     | FINDMOD,  FMOD    | MERGE             | STATUS,   ST
CBSTAT            | FINDUCB,  FINDU   | NAME              | SUMMARY,  SUMM
CLOSE             | GTFTRACE, GTF     | NAMETOKN          | SYSTRACE
COPYDDIR          | INTEGER           | NOTE,     N       | TCBEXIT,  TCBX
COPYDUMP          | IPCS HELP, H      | OPEN              | VERBEXIT, VERBX
COPYTRC           | LIST,     L       | PROFILE,  PROF    | WHERE,    W
CTRACE            | LISTDUMP, LDMP    | RENUM,    REN     |
```

Figure 187. IPCS SUMMARY command

8.8.5 IPCS SUMMARY command

Use the SUMMARY subcommand to display or print dump data associated with one or more specified address spaces.

SUMMARY produces different diagnostic reports depending on the report type parameter, FORMAT, KEYFIELD, JOBSUMMARY, and TCBSUMMARY, and the address space selection parameters, ALL, CURRENT, ERROR, TCBERROR, ASIDLIST, and JOBLIST. Specify different parameters to selectively display the information you want to see.

Note: Installation exit routines can be invoked at the system, address space, and task level for each of the parameters in the SUMMARY subcommand.

The SUMMARY FORMAT command displays task control block (TCB) and other control block information. By issuing the MAX DOWN, or M PF8 command the TCB summary will be located.

Figure 188 on page 299 shows the TCB summary that can be located at the end of an IPCS summary format report. By reviewing the data in the *CMP* field, we see that *TCB 007FD588* has a non-zero CMP field that reflects the 44**0F4**4000 abend.

```
                               * * * *  T C B  S U M M A R Y  * *

JOB SMXC     ASID 0008 ASCB 00FBC580 FWDP 00FBC400 BWDP 00F4E600 PAGE
 TCB AT    CMP      NTC      OTC      LTC      TCB      BACK    PAGE
007FE240 00000000 00000000 00000000 007FDE88 007FF1D8 00000000 000014
007FF1D8 00000000 00000000 007FE240 00000000 007FDE88 007FE240 000018
007FDE88 00000000 007FF1D8 007FE240 007FD588 007FDB70 007FF1D8 000021
007FDB70 00000000 00000000 007FDE88 00000000 007FD588 007FDE88 000024
007FD588 440F4000 02000000 00000000 00000000 00000000 007FBFB8 000026
```

Figure 188. Summary format display - TCB summary

By issuing a `Find "TCB: 007FD588" prev` command, we will be taken to the failing TCB data in the Summary Format display. From this point we want to locate the *RTM2WA* area. This can contain information that in many cases identifies the failing program.

The PSW is still the major reference point and by issuing the following command from the command line:

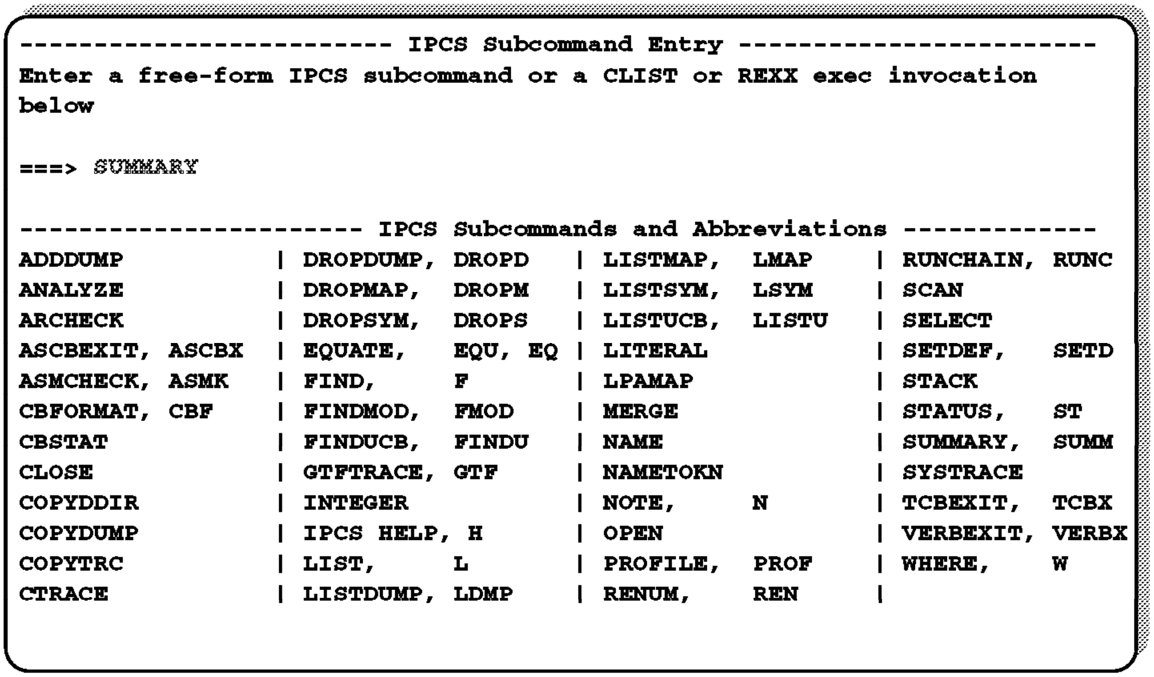
```
   IP WHERE 02CC5BCC.
```

You can see in which load module the failure occurred and where that module was located as follows:

```
 ASID(X'0008') 02CC5BCC. IGWLHHLS+02DBCC IN EXTENDED PLPA
```

Figure 189. IPCS WHERE command display

IPCS VERBX LOGDATA Command



```
------------------------- IPCS Subcommand Entry -----------------------
Enter a free-form IPCS subcommand or a CLIST or REXX exec invocation
below

===> SUMMARY

---------------------- IPCS Subcommands and Abbreviations -------------
ADDDUMP          | DROPDUMP, DROPD   | LISTMAP,  LMAP    | RUNCHAIN, RUNC
ANALYZE          | DROPMAP,  DROPM   | LISTSYM,  LSYM    | SCAN
ARCHECK          | DROPSYM,  DROPS   | LISTUCB,  LISTU   | SELECT
ASCBEXIT, ASCBX  | EQUATE,   EQU, EQ | LITERAL           | SETDEF,   SETD
ASMCHECK, ASMK   | FIND,     F       | LPAMAP            | STACK
CBFORMAT, CBF    | FINDMOD,  FMOD    | MERGE             | STATUS,   ST
CBSTAT           | FINDUCB,  FINDU   | NAME              | SUMMARY,  SUMM
CLOSE            | GTFTRACE, GTF     | NAMETOKN          | SYSTRACE
COPYDDIR         | INTEGER           | NOTE,     N       | TCBEXIT,  TCBX
COPYDUMP         | IPCS HELP, H      | OPEN              | VERBEXIT, VERBX
COPYTRC          | LIST,     L       | PROFILE,  PROF    | WHERE,    W
CTRACE           | LISTDUMP, LDMP    | RENUM,    REN     |
```

Figure 190. IPCS BERBX LOGDATA command

8.8.6 IPCS BERBX LOGDATA command

Specify the LOGDATA verb name on the VERBEXIT subcommand to format the logrec buffer records that were in storage when the dump was generated. LOGDATA locates the logrec records in the logrec recording buffer and invokes the EREP program to format and print the logrec records. The records are formatted as an EREP detail edit report.

Use the LOGDATA report to examine the system errors that occurred just before the error that caused the dump to be requested.

Another valuable source of diagnostic information in the dump is the system error log entries which are created for all hardware and software error conditions. To review these records the VERBX LOGDATA command can be used and the last record(s) should relate to the abend. This is not always the case, but reviewing this data from the last entry and moving backwards in time can often present information that relates to the problem or may indicate what the cause was. This may indicate a hardware or software error. In our case the logdata does include records for our problem and is representative of data already found. Figure 191 on page 301 shows the start of the last error log entry displayed.

```
TYPE:  SOFTWARE RECORD      REPORT:  SOFTWARE EDIT REPORT       DAY.
       (SVC 13)                                REPORT DATE: 103.99
FORMATTED BY: IEAVTFDE  HBB6601                ERROR DATE: 103.99
                            MODEL:   9021                   HH:MM:SS
                            SERIAL:  060143             TIME: 17:21.42

JOBNAME: MSTJCL00   SYSTEM NAME:
ERRORID: SEQ=00080  CPU=0000  ASID=0008  TIME=17:21:42.3

SEARCH ARGUMENT ABSTRACT

  PIDS/5695DF115 RIDS/IGWLHHLS#L RIDS/IGWLGMOT AB/S00F4 PRCS/00000024
  REGS/0C7E8 RIDS/IGWLHERR#R

  SYMPTOM            DESCRIPTION
  -------            -----------
  PIDS/5695DF115     PROGRAM ID: 5695DF115
  RIDS/IGWLHHLS#L    LOAD MODULE NAME: IGWLHHLS
  RIDS/IGWLGMOT      CSECT NAME: IGWLGMOT
  AB/S00F4           SYSTEM ABEND CODE: 00F4
  PRCS/00000024      ABEND REASON CODE: 00000024
  REGS/0E00C         REGISTER/PSW DIFFERENCE FOR R0E: 00C
  REGS/0C7E8         REGISTER/PSW DIFFERENCE FOR R0C: 7E8
  RIDS/IGWLHERR#R    RECOVERY ROUTINE CSECT NAME: IGWLHERR

  OTHER SERVICEABILITY INFORMATION

    RECOVERY ROUTINE LABEL:  IGWFRCSD
    DATE ASSEMBLED:          02/18/94
    MODULE LEVEL:             NONE

  SERVICEABILITY INFORMATION NOT PROVIDED BY THE RECOVERY ROUTINE

    SUBFUNCTION

  TIME OF ERROR INFORMATION

  PSW: 075C2000 82CC5190   INSTRUCTION LENGTH: 02 INTERRUPT CODE: 000D
  FAILING INSTRUCTION TEXT: 41F00024 0A0DBF0F D1D44780
```

Figure 191. IPCS VERBX LOGDATA display

The system error log can also be interrogated via a batch utility. The program used to extract this data from either the online error log data set, SYS1.LOGREC, or a historical error log data set is *IFCEREP1*. This program can be used to produce hardware and software failure reports in both a summary and detailed format. Figure 192 shows the JCL required to process a software summary report.

```
//jobcard
//*
//STEP1 EXEC PGM=IFCEREP1,PARM=CARD
//SYSPRINT DD SYSOUT=*
//SERLOG   DD DSN=SYS1.LOGREC,DISP=SHR
//DIRECTWK DD UNIT=SYSDA,SPACE=(CYL,5,,CONTIG)
//EREPPT   DD SYSOUT=(*,DCB=BLKSIZE=133)
//TOURIST  DD SYSOUT=(*,DCB=BLKSIZE=133)
//SYSIN    DD *
PRINT=PS
TYPE=SIE
ACC=N
TABSIZE=512K
ENDPARM
```

Figure 192. IFCEREP1 JCL to produce a detail edit report

If your LOGREC data is stored in a Coupling Facility log stream data set you can use the *IFCEREP1* program to access this. Figure 193 on page 302 shows the JCL that will enable you to produce error log reports from the log stream data set.

```
//jobcard
//*
//EREPLOG  EXEC  PGM=IFCEREP1,REGION=4M,
//         PARM=(’HIST,ACC=N,TABSIZE=512K,PRINT=PS,TYPE=SIE’)
//ACCIN    DD  DSN=SYSPLEX.LOGREC.ALLRECS,
//         DISP=SHR,
//         SUBSYS=(LOGR,IFBSEXIT,’FROM=(1999/125),TO=YOUNGEST’,
//         ’SYSTEM=SC42’),
//         DCB=(RECFM=VB,BLKSIZE=4000)
//DIRECTWK DD UNIT=SYSDA,SPACE=(CYL,5,,CONTIG)
//TOURIST  DD SYSOUT=*,DCB=BLKSIZE=133
//EREPPT   DD SYSOUT=*,DCB=BLKSIZE=133
//SYSABEND DD SYSOUT=*
//SYSIN    DD DUMMY
```

Figure 193. IFCEREP1 JCL to produce a report from the log stream data set

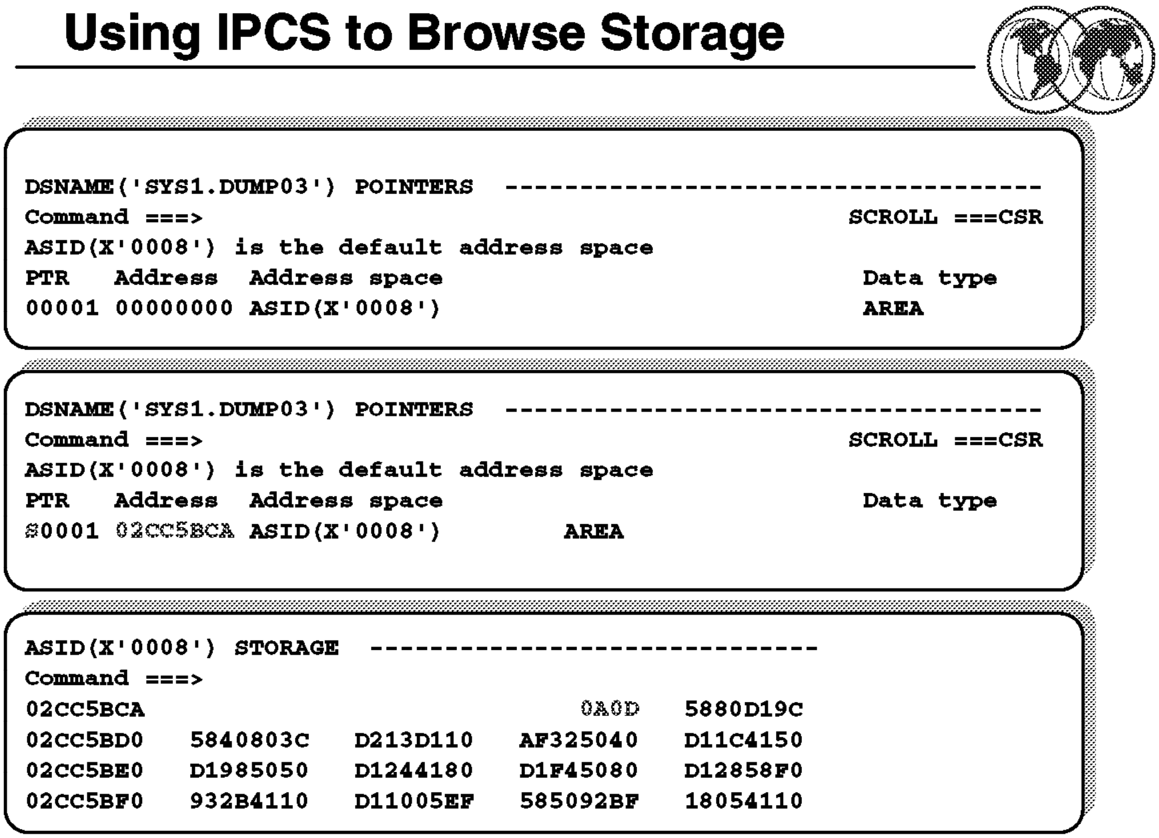
When generating error log reports from log stream data it should be remembered that the log stream data set contains error information for all systems in the sysplex connected to the Coupling Facility. You should use the *SYSTEM* option of the SUBSYS parameter to filter the log stream records. Date and time parameters will also assist with the filtering.

Other information is included in the error log information in the component information. This can assist with isolating the specific product that is being affected and the maintenance level of the module that detected the failure. The maintenance level or service release level is also know as the PTF level, or you might be requested for the replacement modification identifier (RMID). It should be noted that the maintenance level of the failing load module, is not necessarily the maintenance level of the failing CSECT, or module, within the load module.

Figure 194 shows some of the component data that can be located in the system error log.

```
COMPONENT INFORMATION:

    COMPONENT ID:             5695DF115
    COMPONENT RELEASE LEVEL:  1B0
    PID NUMBER:               5695DF1
    PID RELEASE LEVEL:        V1R2
    SERVICE RELEASE LEVEL:    UW04733
    DESCRIPTION OF FUNCTION:  PDSE LATCH SUPPORT
    PROBLEM ID:               IGW00000
    SUBSYSTEM ID:             SMS
```

Figure 194. Error log component information



Using IPCS to Browse Storage

```
DSNAME('SYS1.DUMP03') POINTERS  ----------------------------------
Command ===>                                                SCROLL ===CSR
ASID(X'0008') is the default address space
PTR   Address  Address space                                 Data type
00001 00000000 ASID(X'0008')                                 AREA
```

```
DSNAME('SYS1.DUMP03') POINTERS  ----------------------------------
Command ===>                                                SCROLL ===CSR
ASID(X'0008') is the default address space
PTR   Address  Address space                                 Data type
S0001 02CC5BCA ASID(X'0008')        AREA
```

```
ASID(X'0008') STORAGE  ------------------------------
Command ===>
02CC5BCA                            0A0D   5880D19C
02CC5BD0   5840803C   D213D110   AF325040   D11C4150
02CC5BE0   D1985050   D1244180   D1F45080   D12858F0
02CC5BF0   932B4110   D11005EF   585092BF   18054110
```

Figure 195. Using IPCS to browse storage

8.8.7 Using IPCS to browse storage

Another function of IPCS is the ability to browse storage locations with the dump. Although you have identified the failing instruction text and PSW in many of the displays you have reviewed there will be many times when you will need to look at storage locations. This is achieved by select the **BROWSE** option from the IPCS primary option menu. The next panel will identify the current dump data set, and you can change this if necessary. You will usually enter this function after reviewing other information in the dump, so the requirement to change the dump data set will be unlikely.

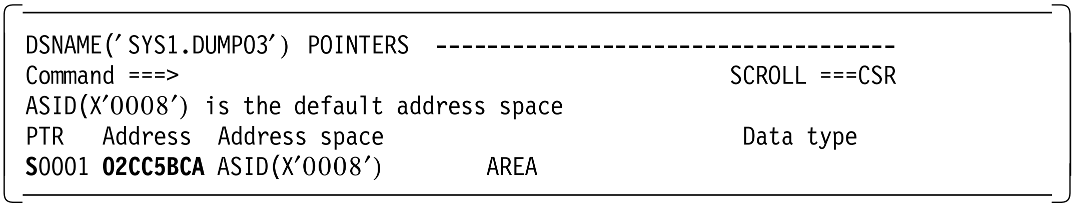
Figure 196 shows the browse panel.

```
DSNAME('SYS1.DUMP03') POINTERS  ----------------------------------
Command ===>                                                SCROLL ===CSR
ASID(X'0008') is the default address space
PTR   Address  Address space                                 Data type
00001 00000000 ASID(X'0008')                                 AREA
```

Figure 196. Browse storage selection screen

From this panel we can overtype the *Address* field with the address specified in the PSW, and **S**elect in the PTR field.

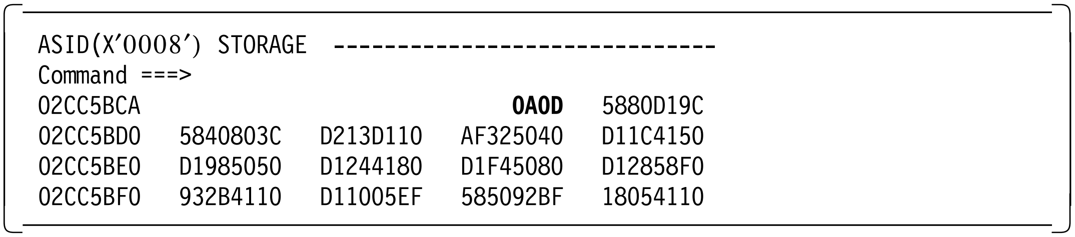
Figure 197 on page 304 shows the browse panel with *S* in the PTR field and the address *02CC5BCA* specified, which is the PSW address minus the instruction length (2).



```
DSNAME('SYS1.DUMP03') POINTERS ------------------------------------
Command ===>                                              SCROLL ===CSR
ASID(X'0008') is the default address space
PTR   Address  Address space                               Data type
S0001 02CC5BCA ASID(X'0008')        AREA
```

Figure 197. Selecting storage location for display

Figure 198 shows the storage at location *02CC5BCA*.



```
ASID(X'0008') STORAGE ------------------------------
Command ===>
02CC5BCA                                0A0D   5880D19C
02CC5BD0   5840803C   D213D110   AF325040   D11C4150
02CC5BE0   D1985050   D1244180   D1F45080   D12858F0
02CC5BF0   932B4110   D11005EF   585092BF   18054110
```

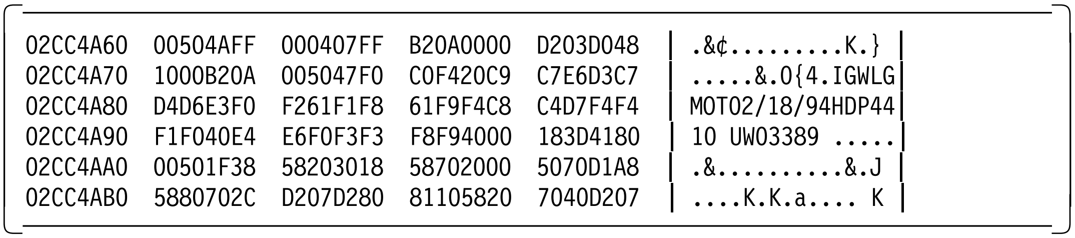
Figure 198. Display of storage at location 02CC5BCA

Figure 198 is showing the storage starting at address *02CC5BCC* minus the instruction length, 2. The address we are looking at is *02CC5BCA* which identifies the failing instruction as *0A0D*. Instruction *0A* is the supervisor call, or SVC instruction. *0D* identifies the SVC number, which in this case is *13*. The hexadecimal value *0D* is equal to *13* decimal.

Figure 182 on page 293 and Figure 191 on page 301 both had a line displayed which identified the `failing instruction text:` 41F00024 0A0D5880 D19C5840 which if compared to the storage displayed in Figure 198 will be found to be the same, 0A0D5880 D19C5840.

SVC 13 is the abend SVC, and in this example we can see that the program itself has taken the abend, as coded. An error condition has occured that has caused a branch to this abend instruction.

Scrolling backwards through the dump, and reviewing the data presented on the right-hand side of the screen leads us to the start of the module, and also shown will be the PTF level of that module, as well as the link-edit date. This can be useful when searching the problem databases or reporting the problem to IBM. The different types of SVCs are documented in *OS/390 MVS Diagnosis Reference*, SY28-1084.



```
02CC4A60  00504AFF  000407FF  B20A0000  D203D048  | .&¢.........K.} |
02CC4A70  1000B20A  005047F0  C0F420C9  C7E6D3C7  | .....&.0{4.IGWLG|
02CC4A80  D4D6E3F0  F261F1F8  61F9F4C8  C4D7F4F4  | MOT02/18/94HDP44|
02CC4A90  F1F040E4  E6F0F3F3  F8F94000  183D4180  | 10 UW03389 .....|
02CC4AA0  00501F38  58203018  58702000  5070D1A8  | .&..........&.J |
02CC4AB0  5880702C  D207D280  81105820  7040D207  | ....K.K.a.... K |
```

Figure 199. Program eyecatcher in storage browse

IBM Problem Databases

- Analyzing the 0F4 abend
  - Examine Messages and Codes
  - Look for following information
    - ABENDxxx or ABENDSxxx
    - MSGxxxxx
    - RCxx
    - RSNxxxxx
- Search IBM databases

Figure 200. Searching IBM problem databases

8.9 Searching IBM problem databases

At this point in time we have evaluated the available diagnostic data from the previous visuals. Look in *OS/390 MVS Systems Codes*, GC28-1780, to find the meaning of the *0F4* abend. The explanation for the 0F4 abend follows:

```
Explanation:  An error occurred in DFSMSdfp support.

Source: DFSMSdfp

System Action:  Prior to the ABEND error occurring, a return code was
placed in the general register 15 and a reason code in general register
0. An SVC dump has been taken unless the failure is in IGWSPZAP
where register 15 contains 10. The DFSMSdfp recovery routines retry to
allow the failing module to return to its caller.
See DFSMS/MVS DFSMSdfp Diagnosis Guide for return code information.

Programmer Response:  System error. Rerun the job.

System Programmer Response:  If the error recurs and the program is not
in error, search problem reporting data bases for a fix for the
problem. If no fix exists, contact the IBM Support Center.
Provide the JCL, the SYSOUT output for the job, and the logrec data set
error record.
```

Note: The *OS/390 MVS System Messages and Systems Codes* manuals should always be your first reference point. These manuals contain important information that could solve your problem and give you suggestions regarding the possible causes and resolutions.

We know that the abend details are:

```
LOAD MODULE NAME: IGWLHHLS   - Maintenance level UW04733
CSECT NAME:       IGWLGMOT   - Maintenance level UW03389
SYSTEM ABEND CODE: 00F4
ABEND REASON CODE: 00000024
RSN=12088C01
```

This information can be used to build the IBM problem database search arguments. The search arguments should use the following formats:

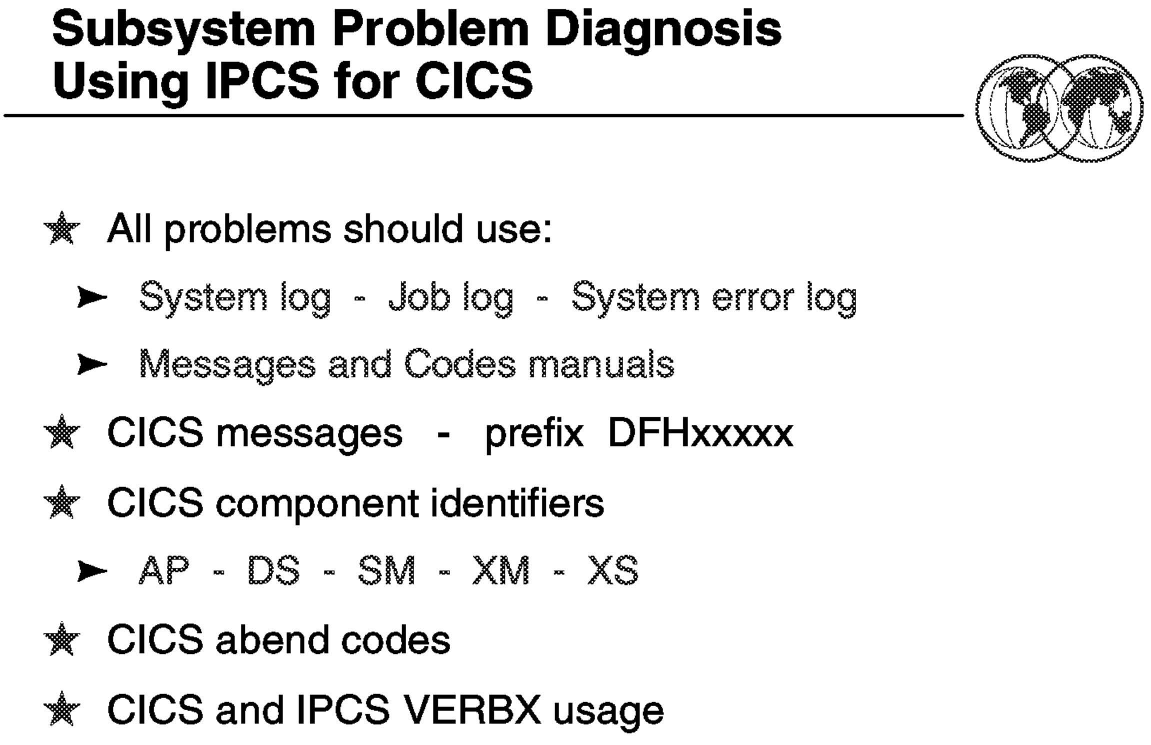
Abend: - The format should be ABEND*xxx* or ABENDS*xxx* where *xxx* is the abend code.

Messages - The format should be MSG*xxxxxxx* where *xxxxxxx* is the message code.

Return and Reason Codes - The format should be RC*xx* where *xx* is the reason or return code. A reason code alternative is:

Reason Codes - The format can be RSN*xxxxx..* where *xxxxx* is the reason code.

These are the recommended formats to be used when querying the problem database or reporting problems. These are not the only formats that are used, and some creativity and imagination can assist with expanding your search. These search arguments are also called a symptom string. If the problem being diagnosed was already reported and the symptoms entered in the database, the search will produce a match.



Subsystem Problem Diagnosis Using IPCS for CICS

- All problems should use:
  - System log - Job log - System error log
  - Messages and Codes manuals
- CICS messages - prefix DFHxxxxx
- CICS component identifiers
  - AP - DS - SM - XM - XS
- CICS abend codes
- CICS and IPCS VERBX usage

Figure 201. Subsystem problem diagnosis using IPCS

8.10 Subsystem problem diagnosis using IPCS

The techniques we have been discussing will assist with most problem analysis, but there are facilities incorporated into IPCS that will assist with the analysis of subsystem problems.

8.10.1 CICS problem diagnosis

Problem diagnosis for CICS, CICSPlex/SM, DB2, IMS, Language Environment, and MQ are just some of the products that can be analyzed using IPCS.

We are unable, within the scope of this document, to provide in-depth solutions to these systems, and the many that I have not mentioned, but we will attempt to provide you with some diagnostic processes that will hopefully assist you with problem determination for these products.

Note: The first reference points for all problems are:

- System log
- Job log
- System error log
- Message and codes manuals

The next sections provide you with some fundamental CICS problem determination techniques.

8.10.1.1 CICS messages

CICS messages consist of the prefix DFH followed by a two-letter component identifier (cc), and a four-digit message number (nnnn). The component identifier shows the domain or the component which issues the message. Some examples of the component identifier are:

- AP - The application domain
- DS - The dispatcher domain
- SM - The storage manager domain
- XM - The transaction manager
- XS - The CICS security component

Thus, the CICS message DFHAP0002 is issued from the application domain, identified by the two-character identifier AP.

8.10.1.2 CICS abend codes

An abend code indicates the cause of an error that may have been originated by CICS or by a user program. For most of the abend codes described, a CICS transaction dump is provided at abnormal termination.

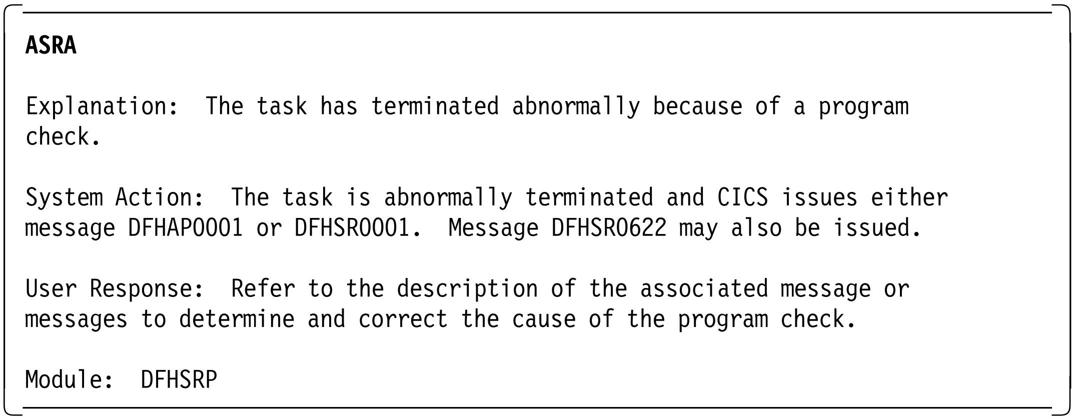
All CICS transaction abend codes abcode are four-character alphanumeric codes of the form *Axxy*, where:

Where:

xx Is the two-character code assigned by CICS to identify the module that detected an error.

y Is the one-character alphanumeric code assigned by CICS.

The description of CICS transaction abend code ASRA is shown in Figure 202.



ASRA

Explanation: The task has terminated abnormally because of a program check.

System Action: The task is abnormally terminated and CICS issues either message DFHAP0001 or DFHSR0001. Message DFHSR0622 may also be issued.

User Response: Refer to the description of the associated message or messages to determine and correct the cause of the program check.

Module: DFHSRP

Figure 202. CICS abend code - ASRA

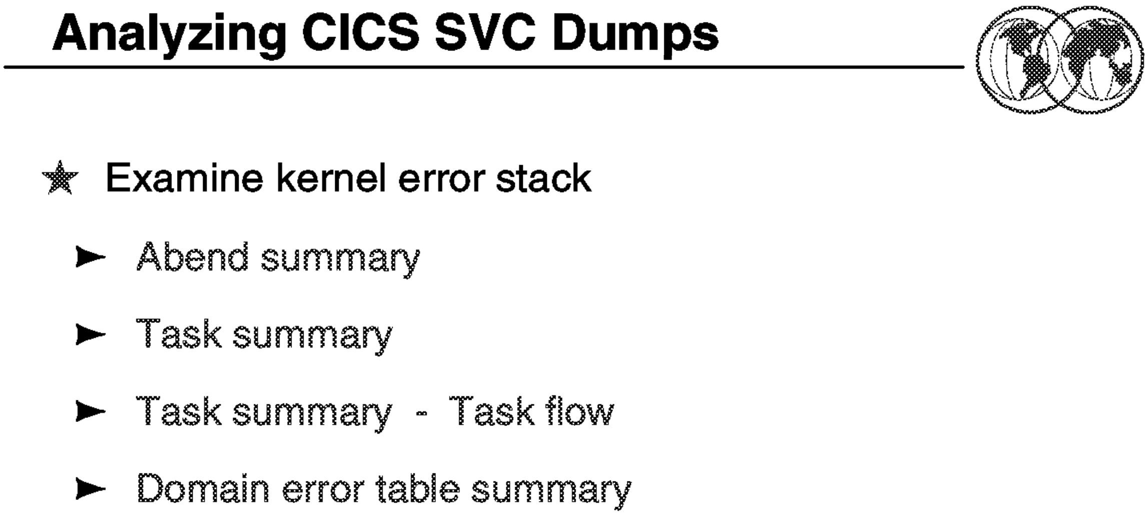
Each release of CICS and CICS/Transaction Server have their own IPCS formatting load modules. The naming convention for these load modules is DFHPD*xxx*, where *xxx* identifies the CICS release, for example:

```
CICS 4.1    - DFHPD410
CICS/TS 1.1 - DFHPD510
CICS/TS 1.2 - DFHPD520
CICS/TS 1.3 - DFHPD530
```

Most commands associated with CICS problem diagnosis using IPCS will be entered from the IPCS Subcommand Entry panel, which is **Option 6 Command** on the IPCS Primary Option Menu. These commands will be prefixed by *VERBX* and the relevant CICS module *DFHPDxxx*, followed by the parameter, for example:

```
VERBX DFHPD530 parameter
```

Many parameters have several levels of complexity. For example, level **1**, might be a summary, level **2**, might be an expanded view, whereas, level **3** may contain both level 1 and 2 data.



Analyzing CICS SVC Dumps

- Examine kernel error stack
  - Abend summary
  - Task summary
  - Task summary - Task flow
  - Domain error table summary

Figure 203. Analyzing CICS SVC dumps

8.10.2 Analyzing CICS SVC dumps

The best place to start when analyzing a CICS SVC dump is with the kernel error stack. This stores information about the last 50 errors that have occurred within the CICS subsystem. This table will only hold 50 errors, so if you have a system that is generating excessive errors, that in turn percolate, this can fill up, and the oldest entries will be discarded. In the majority of cases this should not be a problem, as the kernel error stack will hopefully hold few error identifiers and their related diagnostic information.

The kernel stack entries contain information that has been saved by the kernel on behalf of programs and subroutines on the kernel linkage stack. If you refer the problem to the IBM Support Center, they might need to use the stack entries to find the cause of the failure.

Figure 204 on page 311 shows the first section of the kernel error stack display and summarizes the error condition that caused the dump to be taken.

```
* * * * * CICS 5.3.0 - IPCS EXIT * * * * *
CICS530 OPERANDS:

KE=1

=== SUMMARY OF ACTIVE ADDRESS SPACES

     ASID(hex):         JOBNAME:
     0282               MYCICS01

-- DFHPD0121I FORMATTING CONTROL BLOCKS FOR JOB CICSSA05
DUMPID:   1/0005

DUMPCODE: XM0002

DATE/TIME:18/02/99 09:27:51 (LOCAL)

MESSAGE:  DFHXM0002 CICSSA05 A severe error (code X′100A′) has occured
                                                         in DFHXMIQ
SYMPTOMS: PIDS/565501800 LVLS/530 MS/DFHXM0002 RIDS/DFHXMIQ PTFS/ESA530

TITLE:    (None)

CALLER:   (None)

ASID:     X′0111′
```

Figure 204. CICS kernel error stack display - level 1 - abend summary

Figure 205 shows the next section of the kernel error stack display which identifies all tasks active in the CICS environment at the time the dump was taken, and more importantly it identifies the ****Running**** task, at the time of the abend.

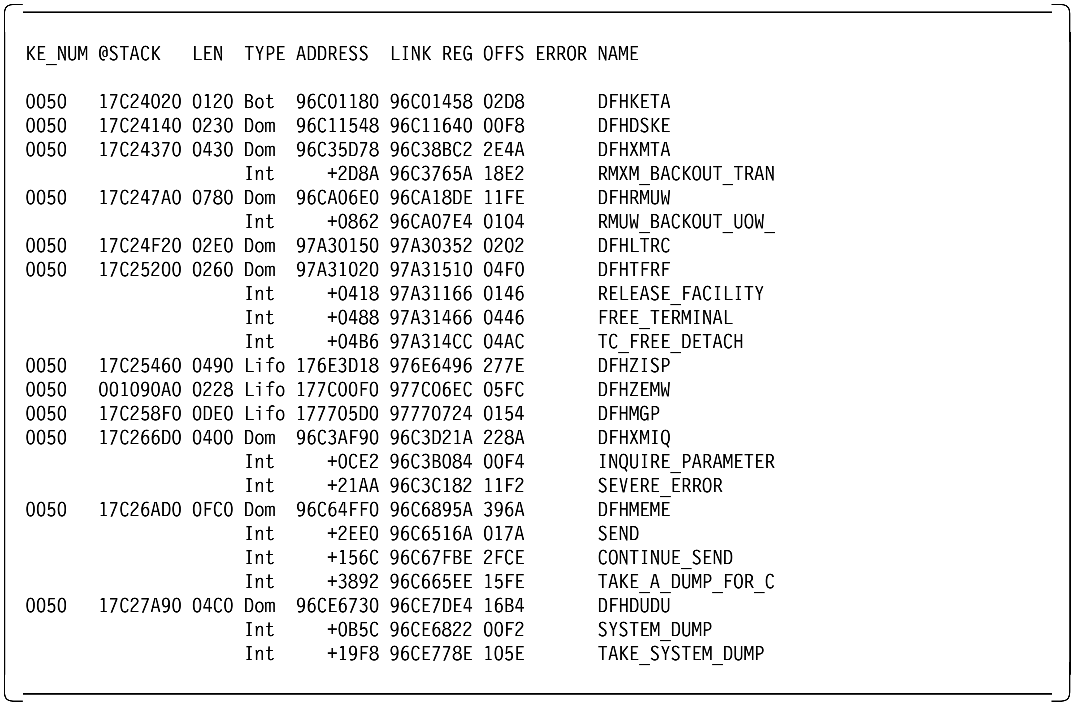
```
KE_NUM KE_TASK  STATUS       TCA_ADDR TRAN_# TRANSID DS_TASK  KE_KTCB  E

0001   16D32C80 KTCB Step    00000000                00000000 16D6C020
0002   16D32900 KTCB QR      00000000                1733B000 16D6F008
0003   16D32580 KTCB RO      00000000                1733D000 16D6E010
0004   16D32200 KTCB FO      00000000                1733F000 16D6D018
0005   16D4BC80 Not Running  00000000                1734A080 16D6E010
0006   16D4B900 Not Running  17485680 00037  CSHQ    1734A280 16D6F008
0007   16D4B580 KTCB         00000000                17374000 1738D008
0008   16D4B200 Not Running  00000000                173E8080 16D6F008
0009   16D64C80 KTCB         00000000                1738E000 17391008
000A   174D9080 Unused
000C   174D9400 Unused
000E   17C5B080 Not Running  00054680 00006  CSSY    17389580 16D6F008
 .
 .
0050   17BFF080 ***Running** 00056680 00128  ABC1    17396780 16D6F008
0051   17BFF400 Unused
0052   17BFF780 Unused
0053   17BFFB00 Unused
0054   17C3C080 Not Running  17486680 00066  ABC2    17396880 16D6F008
0055   17C3C400 Not Running  17484680 00029  COIE    17396280 16D6F008
0056   17C3C780 Not Running  17484080 00022  COIO    17396180 16D6F008
0057   17C3CB00 Not Running  17483680 00020  CONL    17396080 16D6F008
005A   17C9D780 Not Running  00055680 00019  CSNC    1734A680 16D6F008
```

Figure 205. CICS kernel domain - level 1 - task summary

Figure 205 is an abbreviated version of a CICS task summary list. The most important thing to look for is the ****Running**** task. This is "most likely" the problem task.

The ***Running*** task is transaction *ABC1*, which has TRAN_# *00128* and KE_NUM *0050*.

As we scroll down (*PF8*) the display the next section shows the CICS processing flow of each task. Scroll down until we find the KE_NUM associated with the task(s) we identified in the previous step.



```
KE_NUM @STACK   LEN  TYPE ADDRESS  LINK REG OFFS ERROR NAME

0050   17C24020 0120 Bot  96C01180 96C01458 02D8       DFHKETA
0050   17C24140 0230 Dom  96C11548 96C11640 00F8       DFHDSKE
0050   17C24370 0430 Dom  96C35D78 96C38BC2 2E4A       DFHXMTA
                     Int      +2D8A 96C3765A 18E2       RMXM_BACKOUT_TRAN
0050   17C247A0 0780 Dom  96CA06E0 96CA18DE 11FE       DFHRMUW
                     Int      +0862 96CA07E4 0104       RMUW_BACKOUT_UOW_
0050   17C24F20 02E0 Dom  97A30150 97A30352 0202       DFHLTRC
0050   17C25200 0260 Dom  97A31020 97A31510 04F0       DFHTFRF
                     Int      +0418 97A31166 0146       RELEASE_FACILITY
                     Int      +0488 97A31466 0446       FREE_TERMINAL
                     Int      +04B6 97A314CC 04AC       TC_FREE_DETACH
0050   17C25460 0490 Lifo 176E3D18 976E6496 277E       DFHZISP
0050   001090A0 0228 Lifo 177C00F0 977C06EC 05FC       DFHZEMW
0050   17C258F0 0DE0 Lifo 177705D0 97770724 0154       DFHMGP
0050   17C266D0 0400 Dom  96C3AF90 96C3D21A 228A       DFHXMIQ
                     Int      +0CE2 96C3B084 00F4       INQUIRE_PARAMETER
                     Int      +21AA 96C3C182 11F2       SEVERE_ERROR
0050   17C26AD0 0FC0 Dom  96C64FF0 96C6895A 396A       DFHMEME
                     Int      +2EE0 96C6516A 017A       SEND
                     Int      +156C 96C67FBE 2FCE       CONTINUE_SEND
                     Int      +3892 96C665EE 15FE       TAKE_A_DUMP_FOR_C
0050   17C27A90 04C0 Dom  96CE6730 96CE7DE4 16B4       DFHDUDU
                     Int      +0B5C 96CE6822 00F2       SYSTEM_DUMP
                     Int      +19F8 96CE778E 105E       TAKE_SYSTEM_DUMP
```

Figure 206. CICS kernel domain task summary - task flow KE_NUM=0050

Figure 206 shows us information about the processing flow of transaction ABC1. When the failure occurred one of the following was indicated:

- INQUIRE_PARAMETER
- SEVERE_ERROR
- DFHMEME
- SEND
- CONTINUE_SEND
- TAKE_A_DUMP_FOR_CALLER
- DFHDUDU
- SYSTEM_DUMP
- TAKE_SYSTEM_DUMP

Often the *ERROR* column will indicate a failure with *YES* next to the failure. In this case it does not but the *SEVERE_ERROR* following the *INQUIRE_PARAMETER* points us in the right direction.

Figure 207 on page 313 is at the end of the *KE=1* display and displays all errors in the kernel error stack, with a maximum of 50 errors records being kept.

```
==KE: KE Domain Error Table Summary

ERR_NUM  ERR_TIME KE_NUM ERROR TYPE           ERR_CODE MODULE   OFFSET
=======  ======== ====== ==========           ======== ======   ======
00000001 09:19:47  0051  PROGRAM_CHECK        0C7/AKEA DFHYC520 000D7A
00000002 09:19:47  0051  TRAN_ABEND_PERCOLATE ---/ASRA DFHSR1   0003C2
00000003 09:19:48  0051  PROGRAM_CHECK        0C4/AKEA -noheda- 0089DC
00000004 09:19:48  0051  ABEND                ---/0999 DFHKERET 000064
00000005 09:23:02  0050  PROGRAM_CHECK        0C7/AKEA DFHYC520 000D7A
00000006 09:23:02  0050  TRAN_ABEND_PERCOLATE ---/ASRA DFHSR1   0003C2
00000007 09:23:02  0050  PROGRAM_CHECK        0C4/AKEA -noheda- 0089DC
00000008 09:23:02  0050  ABEND                ---/0999 DFHKERET 000064
```

Figure 207. Kernel number displayed

Figure 207 shows us the kernel number associated with what we suspect is the failing task, KE_NUM=0050, that abended at 09:23:02. The last entry, ERR_NUM 00000008, although for KE_NUM=0050, is not the record we are interested in, because it is most likely the result of earlier abends, in this case, error numbers 5, 6, and 7.

The first abend in the sequence, error number 5, is what we would usually want to investigate, and errors 6, 7, and 8 are most likely the result of a percolation of the error number 5 condition.

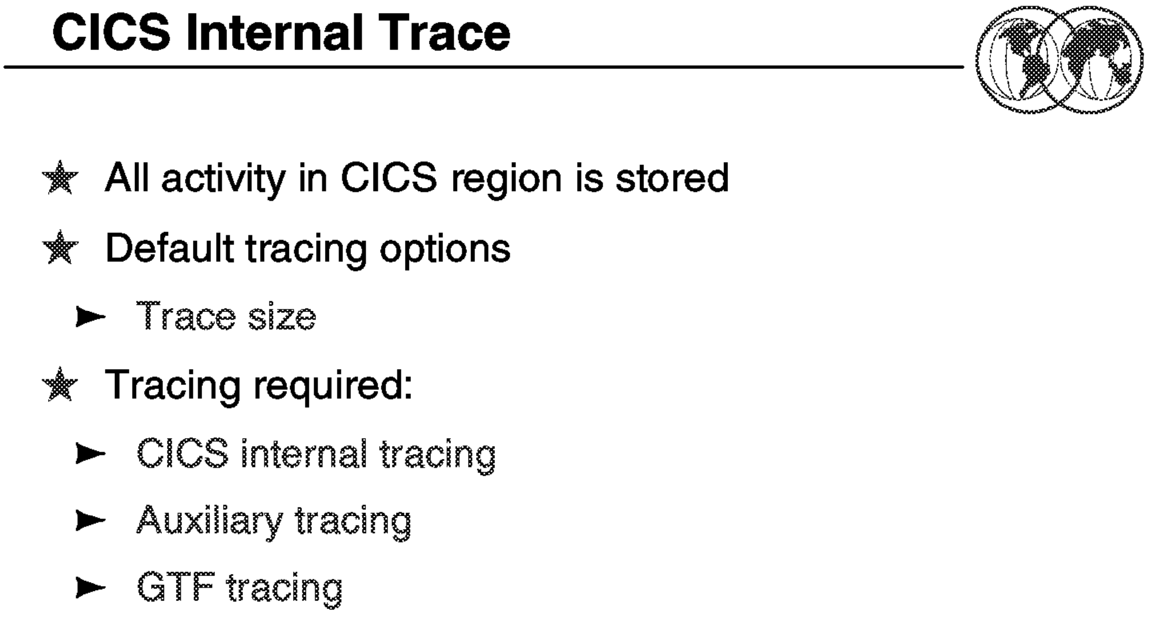
The first abend for KE_NUM=0050 is identified as a *PROGRAM CHECK*, where an *ABEND0C4* was detected in load module *DFHYC520* at offset X′000D7A′. This is reported to CICS as an *AKEA* abend.

Note: If you refer to Figure 204 on page 311 you will notice that the dump was taken for an abend at *09:27:51* and was for a severe error in module DFHXMIQ. The error code was X′100A′. Although there might be a relationship with the errors identified in the kernel error stack for KE_NUM=0050, the time difference of just under 5 minutes between the last recorded error and the dump will need to be checked to verify whether there is a relationship.

8.10.2.1 The kernel error stack - VERBX DFHPDxxx ′KE=2′

Further investigation of errors identified in the kernel error stack can be facilitated by using *VERBX DFHPDxxx ′KE=2′*. This display contains information and abbreviated storage dumps relating to each of these errors. The KE=2 option displays from the top (oldest) to the bottom (newest) entry. To find the start of each error issue the following command:

```
FIND ′ERROR NUMBER:′
```



CICS Internal Trace

- All activity in CICS region is stored
- Default tracing options
  - Trace size
- Tracing required:
  - CICS internal tracing
  - Auxiliary tracing
  - GTF tracing

Figure 208. CICS internal trace

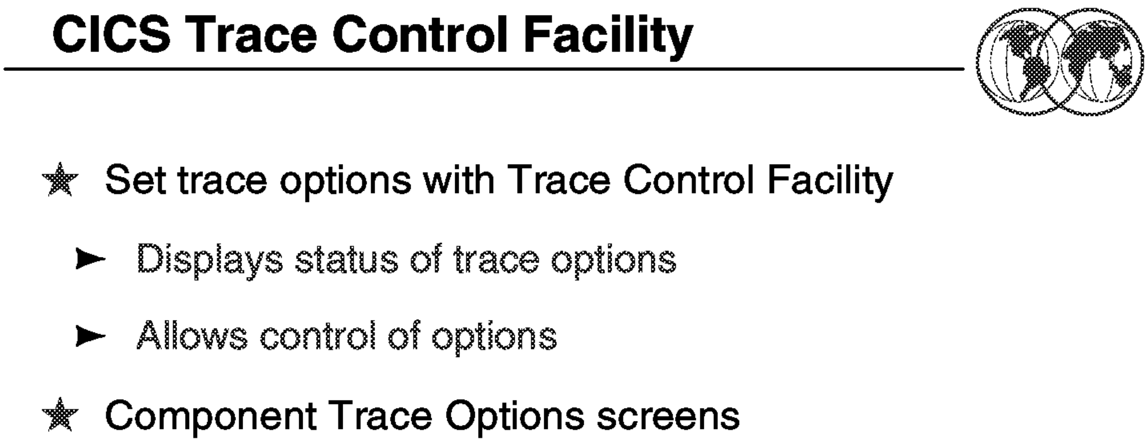
8.10.3 CICS internal trace

Possibly the most useful diagnostic information that can be reviewed in a CICS SVC dump, is the CICS internal trace. A record of all activity within the CICS region is stored. The default tracing options only capture *exception trace* entries. Unfortunately, we usually like to review what preceded the generation of an exception condition, as this is the only way to see what was the cause, not just the result.

Another unfortunate default is the CICS internal trace size which is set to 64K. In a busy CICS region this would store less that one second of trace data. Hardly enough to enable you to review the flow of the transaction that caused the exception condition. We recommend an internal trace size of at least 2500K. This should provide you with sufficient trace information, except in exceptionally busy systems. In a large, high usage environment a 10000K trace table can hold as little a five seconds of trace data.

CICS tracing is performed by the trace domain at predetermined trace points in the CICS code during the regular flow of control. This includes user tracing from applications. Tracing is performed when you turn on CICS internal tracing, auxiliary tracing, and GTF tracing. You can control the type of tracing to suit your needs, except when an exception condition is detected by CICS. CICS always makes an exception trace entry. You cannot turn exception tracing off. Trace points are included at specific points in CICS code; from these points, trace entries can be written to any currently selected trace destination. All CICS trace points are listed in alphanumeric sequence the *CICS User′s Handbook*, SX33-1188.

Level-1 trace points are designed to give you enough diagnostic information to fix “user” errors. Level-2 trace points are situated between the level-1 trace points, and they provide information that is likely to be more useful for fixing errors within CICS code. You probably will not want to use level-2 trace points yourself, unless you are requested to do so by IBM support staff after you have referred a problem to them.



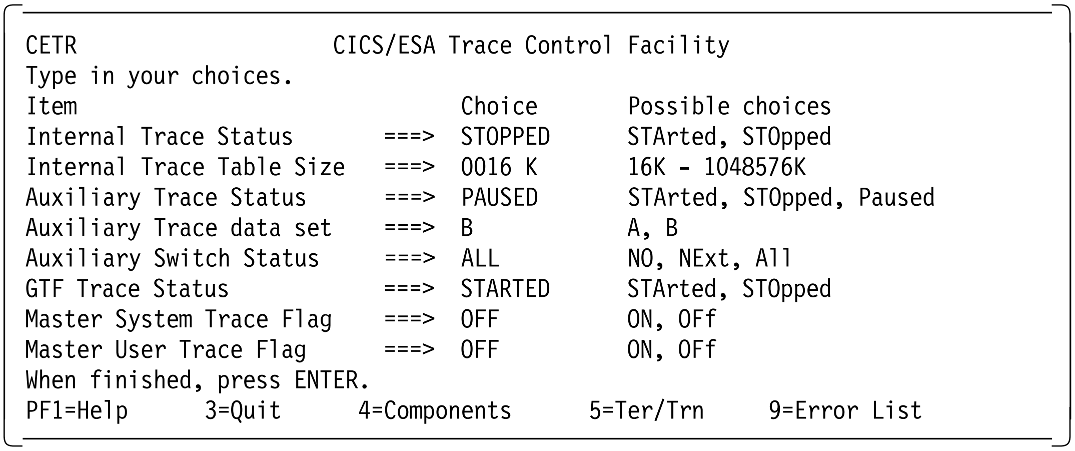
CICS Trace Control Facility

- Set trace options with Trace Control Facility
  - Displays status of trace options
  - Allows control of options
- Component Trace Options screens

Figure 209. CICS Trace Control Facility

8.10.4 CICS Trace Control Facility

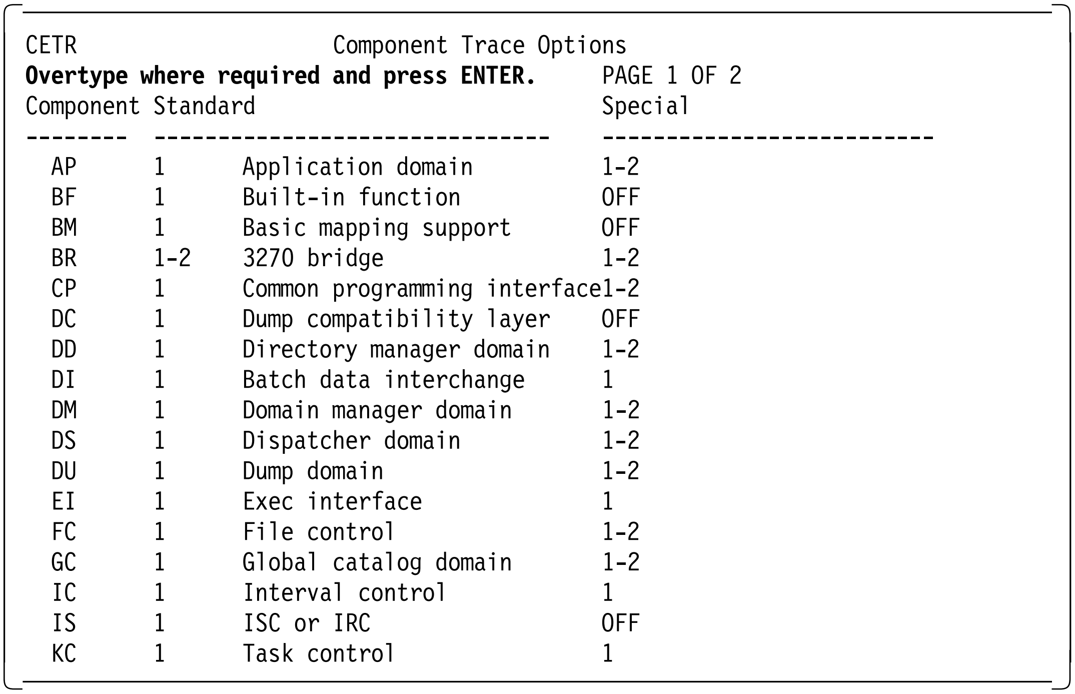
CICS exception tracing is always done by CICS when it detects an exception condition. The sorts of exception that might be detected include bad parameters on a domain call, and any abnormal response from a called routine. The aim is “first failure data capture,” to record data that might be relevant to the exception as soon as possible after it has been detected. The trace options can be set using the CICS/ESA Trace Control Facility *(CETR)* transaction. This will enable you to increase the trace table size and let you control CICS component (domain) trace options. Tracing for all CICS components should be set to *level 1* when collecting diagnostic data. Figure 210 on page 317 shows the CETR panel that can be used to control tracing options.



```
CETR                   CICS/ESA Trace Control Facility
Type in your choices.
Item                              Choice       Possible choices
Internal Trace Status       ===>  STOPPED      STArted, STOpped
Internal Trace Table Size   ===>  0016 K       16K - 1048576K
Auxiliary Trace Status      ===>  PAUSED       STArted, STOpped, Paused
Auxiliary Trace data set    ===>  B            A, B
Auxiliary Switch Status     ===>  ALL          NO, NExt, All
GTF Trace Status            ===>  STARTED      STArted, STOpped
Master System Trace Flag    ===>  OFF          ON, OFf
Master User Trace Flag      ===>  OFF          ON, OFf
When finished, press ENTER.
PF1=Help     3=Quit     4=Components     5=Ter/Trn    9=Error List
```

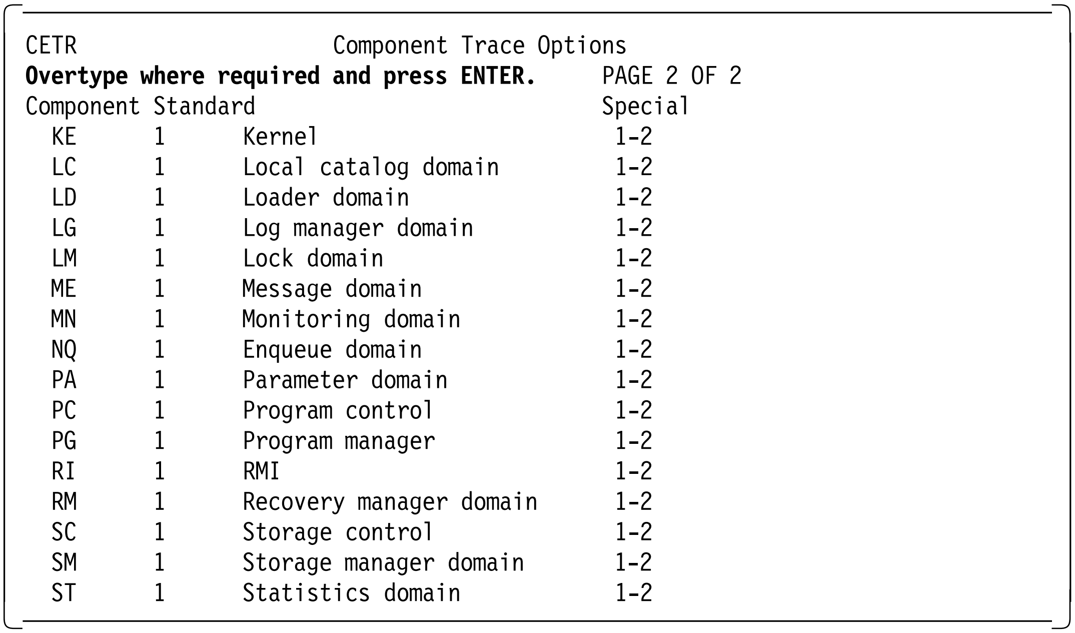
Figure 210. CICS/ESA Trace Control Facility

Figure 211, Figure 212 on page 318, and Figure 213 on page 318 show the screens of the CETR component options. This enables you to set tracing options for each CICS domain component. The description for each component does not appear in the screen and has been included here for your reference.



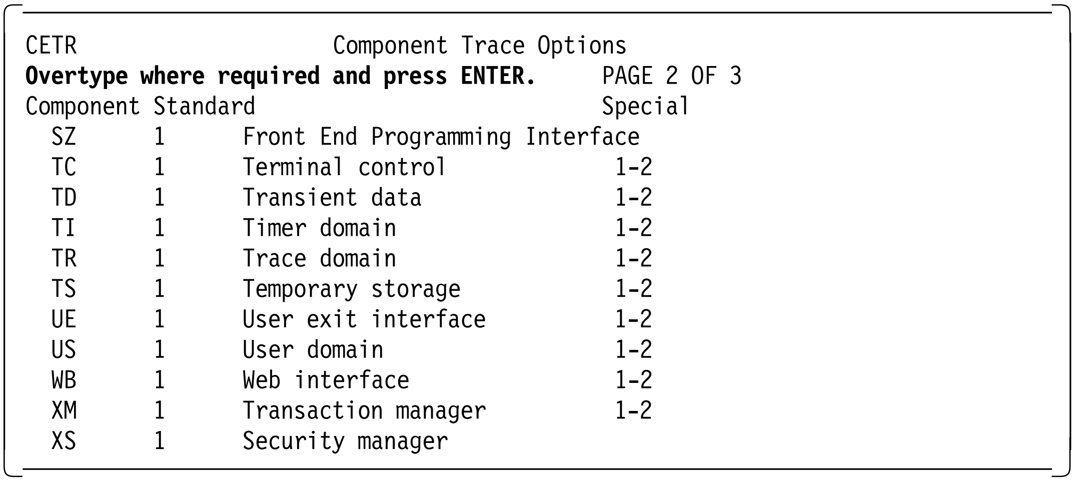
```
CETR                   Component Trace Options
Overtype where required and press ENTER.     PAGE 1 OF 2
Component Standard                           Special
--------  -------------------------------    -------------------------
  AP      1      Application domain          1-2
  BF      1      Built-in function           OFF
  BM      1      Basic mapping support       OFF
  BR      1-2    3270 bridge                 1-2
  CP      1      Common programming interface1-2
  DC      1      Dump compatibility layer    OFF
  DD      1      Directory manager domain    1-2
  DI      1      Batch data interchange      1
  DM      1      Domain manager domain       1-2
  DS      1      Dispatcher domain           1-2
  DU      1      Dump domain                 1-2
  EI      1      Exec interface              1
  FC      1      File control                1-2
  GC      1      Global catalog domain       1-2
  IC      1      Interval control            1
  IS      1      ISC or IRC                  OFF
  KC      1      Task control                1
```

Figure 211. CICS/ESA Trace Control Facility component options



```
CETR                    Component Trace Options
Overtype where required and press ENTER.     PAGE 2 OF 2
Component Standard                           Special
  KE      1      Kernel                       1-2
  LC      1      Local catalog domain         1-2
  LD      1      Loader domain                1-2
  LG      1      Log manager domain           1-2
  LM      1      Lock domain                  1-2
  ME      1      Message domain               1-2
  MN      1      Monitoring domain            1-2
  NQ      1      Enqueue domain               1-2
  PA      1      Parameter domain             1-2
  PC      1      Program control              1-2
  PG      1      Program manager              1-2
  RI      1      RMI                          1-2
  RM      1      Recovery manager domain      1-2
  SC      1      Storage control              1-2
  SM      1      Storage manager domain       1-2
  ST      1      Statistics domain            1-2
```

Figure 212. CICS/ESA Trace Control Facility component options screen 2



```
CETR                    Component Trace Options
Overtype where required and press ENTER.     PAGE 2 OF 3
Component Standard                           Special
  SZ      1      Front End Programming Interface
  TC      1      Terminal control             1-2
  TD      1      Transient data               1-2
  TI      1      Timer domain                 1-2
  TR      1      Trace domain                 1-2
  TS      1      Temporary storage            1-2
  UE      1      User exit interface          1-2
  US      1      User domain                  1-2
  WB      1      Web interface                1-2
  XM      1      Transaction manager          1-2
  XS      1      Security manager
```

Figure 213. CICS/ESA Trace Control Facility component options screen 3

If you have a requirement to capture trace data over a more substantial period of time, or trace the CICS system without taking a dump, you can use the CICS AUXTRACE facility. CICS auxiliary trace entries are directed to one of two auxiliary trace data sets, DFHAUXT and DFHBUXT. These are CICS-owned BSAM data sets, and they must be created before CICS is started. They cannot be redefined dynamically. The amount of data that can be collected is related to the size of the auxiliary trace data sets. You can use the AUXTR system initialization parameter to turn CICS auxiliary trace on or off in the system initialization table. Alternatively, You can select a status of STARTED, STOPPED, or PAUSED for CICS auxiliary trace dynamically using the CETR transaction.

`VERBX DFHPDxxx 'TR=1'` formats an abbreviated trace, which displays one line per trace entry. Whereas, `VERBX DFHPDxxx 'TR=2'` formats an expanded trace, which produces a more comprehensive listing for each instruction.

The size of the data formatted for a *TR=2* request can be excessive, and due to the limitations of your TSO region size, can make it impossible to *MAX* to the end of the trace. Which is usually where you will want to start. To assist with this process you can select only the trace data that you want to review. For example:

From the *KE=1* display you have identified the KE_NUM of the task you want to review. Issuing the following command will let you select only the trace data related to the specified task:

```
VERBX DFHPD530 'TR=2,TRS=<KE_NUM=0050>'
```

Figure 214 shows an abbreviated trace listing. The best way to approach the trace is to *MAX PF8* to the end of the display, and then issue the `FIND *EXC PREV` command. This will locate the last *EXCEPTION* in the trace table. The entries immediately preceding this exception can often identify the problem.

```
00128 QR AP 1710 TFRF  ENTRY RELEASE_FACILITY      NO,ABNORMAL,187A92
00128 QR AP 00E0 MGP   ENTRY 02206                 TERM
00128 QR AP 1700 TFIQ  ENTRY INQUIRE_TERMINAL_FACILITY 00000000 , 000
00128 QR AP 1701 TFIQ  EXIT  INQUIRE_TERMINAL_FACILITY/EXCEPTION
                             NO_TERMINAL
00128 QR XM 100A XMIQ  *EXC* Unexpected_return_code_from_TFIQ_INQUIRE
                             _REQUEST INQUIRE_TRANSACTION,000128C
00128 QR KE 0101 KETI  ENTRY INQ_LOCAL_DATETIME_DECIMAL
```

*Figure 214. Abbreviated trace listing - *EXCeption located*

Application program related trace points can be found in the trace by the *EIP* and *APLI* identifiers. It should also be noted that the trace shows the CICS processing flow of an *ENTRY* and *EXIT* for each function.

Figure 215 shows some *APLI* and *EIP* entries and also displays the *ENTRY* and *EXIT* relationship.

```
AP 00E1 EIP   ENTRY FREEMAIN
SM 0D01 SMMF  ENTRY FREEMAIN             00140448,EXEC,CICS
SM 0D02 SMMF  EXIT  FREEMAIN/OK          USER24 storage at 001404
AP 00E1 EIP   EXIT  FREEMAIN             OK
AP 1949 APLI  EVENT RETURN-FROM-LE/370   Rununit_Termination OK C
AP 1948 APLI  EVENT CALL-TO-LE/370       Thread_Termination
AP 1949 APLI  EVENT RETURN-FROM-LE/370   Thread_Termination OK
AP 1941 APLI  EXIT  START_PROGRAM/EXCEPTION TRANSACTION_ABEND,ASRA
AP 0510 APAC  ENTRY REPORT_CONDITION
AP 1940 APLI  ENTRY START_PROGRAM        DFHTFP,NOCEDF,FULLAPI,SY
AP 00DD TFP   ENTRY TRANSACTION_ABENDED
SM 0301 SMGF  ENTRY FREEMAIN             1734CC14 , 0000006A,17C6
SM 0302 SMGF  EXIT  FREEMAIN/OK
```

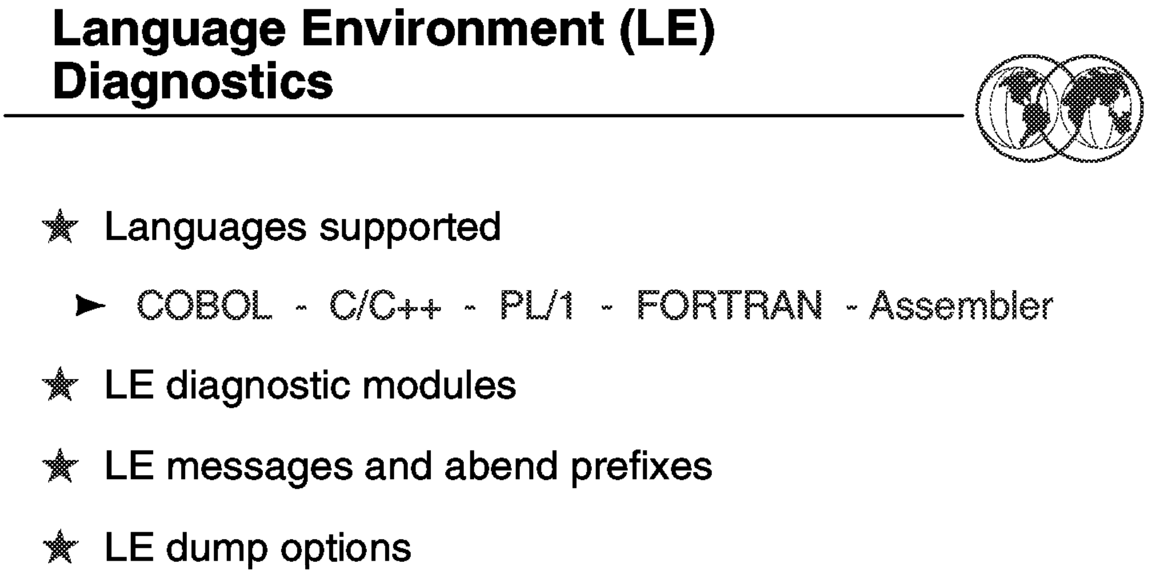
Figure 215. CICS trace showing EIP, APLI, ENTRY, and EXIT

```
XM 100A XMIQ  *EXC* - Unexpected_return_code_from_TFIQ_INQUIRE_TERMINAL_
                   (0000128C)
TASK-00128 KE_NUM-0050 TCB-QR   /008C7C40 RET-97770724 TIME-09:26:04.862
  1-0000  00F80000 000000A1 00000000 00000000  B020E000 04A00000 0178010
    0020  33782800 00000217 C1C1C4D4 68000000  00002800 00000000 000000A
    0040  C1C1C4D4 C3C3C1C1 C4C5D4D6 C4C6C8C3  C9C3E2E3 C2563000 0000600
    0060  000000A5 00000000 00000001 01010101  01010101 01010105 A901011
    0080  000060F1 0000128C C25A0001 01010101  01010101 01010117 174087E
    00A0  01010017 41E0D800 00000000 41E0D800  00000000 00100096 C29B7C9
    00C0  C29D5A96 C29F3600 01010100 00000097  629A5017 BFF08000 0060000
    00E0  00000017 C25E4096 C29AF817 C25A0000  00000017 C2527000
  2-0000  00780000 000000CD 00000000 00000000  B4040000 00000000 0100020
    0020  00000000 00000000 00000000 00000000  00000000 00000000 0000000
    0040  00000000 00000000 00000000 00000000  00000000 00000000 0000000
    0060  00000000 00000000 00000000 00000000  00000000 00000001
```

*Figure 216. Expanded trace listing - *EXCeption entry*

Figure 216 shows one expanded trace entry. These expanded entries show the parameter lists that are being used by CICS and to the right of the display (not shown), is an ASCII “eyecatcher” that can also assist with identifying user data.

CICS SVC dumps contain information about all the domain structures that make up CICS. These domains can be formatted using IPCS and are documented in the *CICS Diagnosis Guide*, LX33-6093. Trace information can be found in the *CICS User's Handbook*, SX33-1188. All can be invaluable in assisting with problem determination.



Language Environment (LE) Diagnostics

- Languages supported
  - COBOL - C/C++ - PL/1 - FORTRAN - Assembler
- LE diagnostic modules
- LE messages and abend prefixes
- LE dump options

Figure 217. Language Environment diagnostic procedures

8.11 Language Environment diagnostic procedures

Language Environment (LE) provides a common run-time environment accross multiple high level languages. These languages include:

- COBOL
- C/C++
- PL/I
- FORTRAN
- Assembler (not HLL)

LE is supported across multiple platforms, OS/390, VM, and VSE.

A run-time environment provides facilities, such as storage control, system time and date functions, error processing, message processing and other system functions to the high-level languages. The run-time library is "called" by the user program to perform these functions. Before LE, each high-level language had its own run-time library, but LE has combined the functionality required by each language into a single run-time environment.

There are two Common Execution Library (CEL) modules that will indicate a failure, but the cause will be elsewhere. The first is *CEEHDSP* which schedules the LE CEEDUMP to be taken. The second module is *CEEPLPKA*, which will always indicate an ABENDU4039 or ABENDU4038 no matter what the original error. Your diagnostic methodology should exclude failures in these two modules.

The LE event handler modules are identified as *CEEExxx* where *xxx* represents the language, as follows:

003 C/C++ Run-time (that is, CEEEV003)

005 COBOL

007 FORTRAN

008 DCE

010 PL/I

012 Debug Tool

The following messages and abend prefixes can assist with problem diagnosis.

CEE Is output by CEL but may be reporting a problem elsewhere

IGZ Is output by COBOL

IBM Is output by PL/I

AFH Is output by FORTRAN

EDC Is output by C/C++

Some common CEL messages that indicate exception (*0Cx*) conditions are:

- CEE3201 = ABEND0C1
- CEE3204 = ABEND0C4
- CEE32xx = ABEND0Cy, where y is the hex equivalent of decimal xx

Message CEE3250 indicates a non exception (0Cx) abend has occured.

Common CEL ABENDS:

U4038 Some "severe" error occurred but no dump was requested.

U4039 Some "severe" error occurred and a dump was requested.

U4083* Backchain in error - only occurs after some other error.

U4087* Error during error processing.

U4093* Error during initialization.

U4094* Error during termination.

The ***** indicates that a reason code is required for this message to be meaningful.

8.11.1.1 LE dump options

To enable you to diagnose problems in an LE environment it is necessary to obtain either a CEEDUMP, and/or a U4039 dump. The run-time options that should be set to request a dump from LE are:

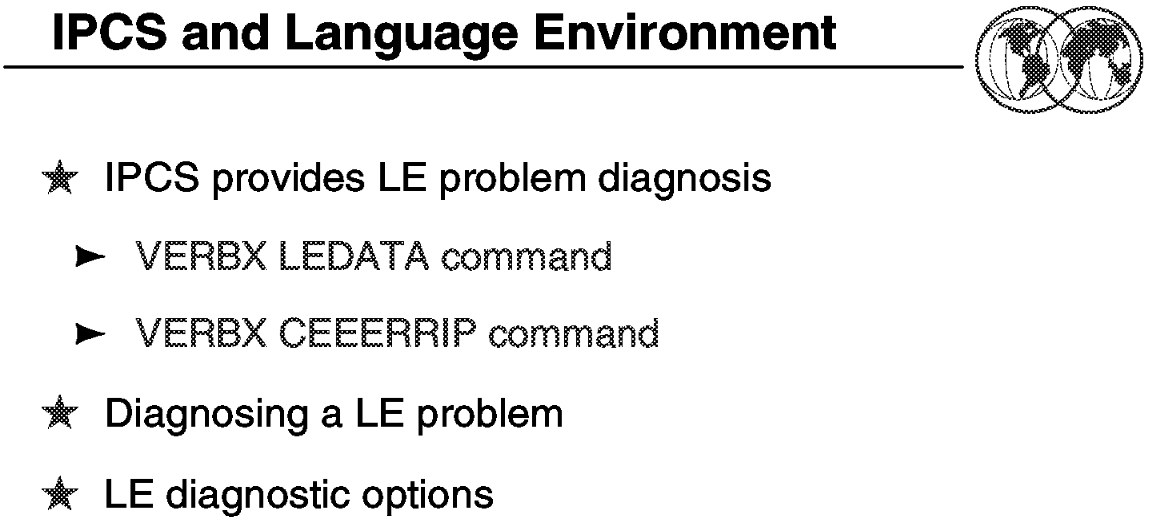
- ABTERMENC(ABEND) - User will see the "original" abend. For example, *0C4*

- TERMTHDACT(UADUMP)

- TRAP(ON)

CEEDUMP output will go to SYSOUT and the system dump is dependant on whether a SYSMDUMP DD card has been included in the JCL. Note that LRECL=4160 must be used for SYSMDUMP.

Setting a SLIP for an LE-handled abend is pointless because the LE abend is reissued at the end of LE termination and clean up processing for the LE environment has already been performed.



IPCS and Language Environment

- IPCS provides LE problem diagnosis
  - VERBX LEDATA command
  - VERBX CEEERRIP command
- Diagnosing a LE problem
- LE diagnostic options

Figure 218. IPCS and Language Environment

8.11.2 IPCS and Language Environment

IPCS provides some facilities to assist with LE problem diagnosis. The IPCS command `VERBX LEDATA` or `VERBX CEEERRIP` shows the LE run-time options and general information about your LE environment at the time of the failure.

CEEERRIP is the LE diagnostic module that is used to format the dump data. `VERBX CEEERRIP 'CEEDUMP'` will format the traceback information, as shown in Figure 219 on page 324.

```
Information for enclave main

Information for thread 8000000000000000

Traceback:
DSA Addr Program Unit PU Addr   PU Offset  Entry    E Addr   Status
00022460 CEEHSDMP     12B7F950  -0001FE3A  CEEHSDMP 12B7F950 call
00020018 CEEHDSP      12B27138  +000026A4  CEEHDSP  12B27138 call
000223C0              12B00B80  +00000062  func_#1  12B00B80 exception
00022320              12B00A00  +0000005E  func_#2  12B00A00 call
00022280              12B00880  +0000005E  func_#3  12B00880 call
000221E0              12B005E8  +00000066  main     12B005E8 call
000220C8              12D5BE0E  +000000B4  EDCZMINV 12D5BE0E call
00022018 CEEBBEXT     0000E690  +0000013C  CEEBBEXT 0000E690 call
```

Figure 219. IPCS display of LE traceback information

The VERBX CEEERRIP display is shown in Figure 220.

```
LAST WHERE SET          Override   OPTIONS
************************************************************************
INSTALLATION DEFAULT    OVR        ABPERC(NONE)
PROGRAM INVOCATION      OVR        ABTERMENC(ABEND)
INSTALLATION DEFAULT    OVR      NOAIXBLD
INSTALLATION DEFAULT    OVR        ALL31(OFF)
INSTALLATION DEFAULT    OVR        ANYHEAP(00016384,00008192,ANY  ,FREE)
INSTALLATION DEFAULT    OVR      NOAUTOTASK
INSTALLATION DEFAULT    OVR        BELOWHEAP(00008192,00004096,FREE)
INSTALLATION DEFAULT    OVR        CBLOPTS(ON)
INSTALLATION DEFAULT    OVR        CBLPSHPOP(ON)
INSTALLATION DEFAULT    OVR        CBLQDA(ON)
INSTALLATION DEFAULT    OVR        CHECK(ON)
INSTALLATION DEFAULT    OVR        COUNTRY(US)
INSTALLATION DEFAULT    OVR        DEBUG
INSTALLATION DEFAULT    OVR        DEPTHCONDLMT(00000010)
INSTALLATION DEFAULT    OVR        ENVAR("")
INSTALLATION DEFAULT    OVR        ERRCOUNT(00000020)
INSTALLATION DEFAULT    OVR        ERRUNIT(00000006)
INSTALLATION DEFAULT    OVR        FILEHIST
DEFAULT SETTING         OVR      NOFLOW
INSTALLATION DEFAULT    OVR        HEAP(00032768,00032768,ANY  ,
                                        KEEP,00008192,00004096)
INSTALLATION DEFAULT    OVR        HEAPCHK(OFF,00000001,00000000)
```

Figure 220. IPCS LE display of run-time options (partial)

8.11.2.1 Finding the failing CSECT name in LE

Getting the name of the failing csect (function), or any LE-enabled CSECT can be performed as follows:

1. Get address (second word) from MCH_PSW.
2. Select Option 1 (Browse) in IPCS.
3. L xxxxxxxx (where xxxxxxxx is the addr).
4. F CEE prev.
5. Back up five bytes to the 47F0xxxx instruction. This is the beginning of the CSECT/function.
6. Add the value at offset X′0C′ to the module address.

7. Go to that location.
8. Add X′20′ bytes. This is a two-byte prefixed string (length) with the function name.

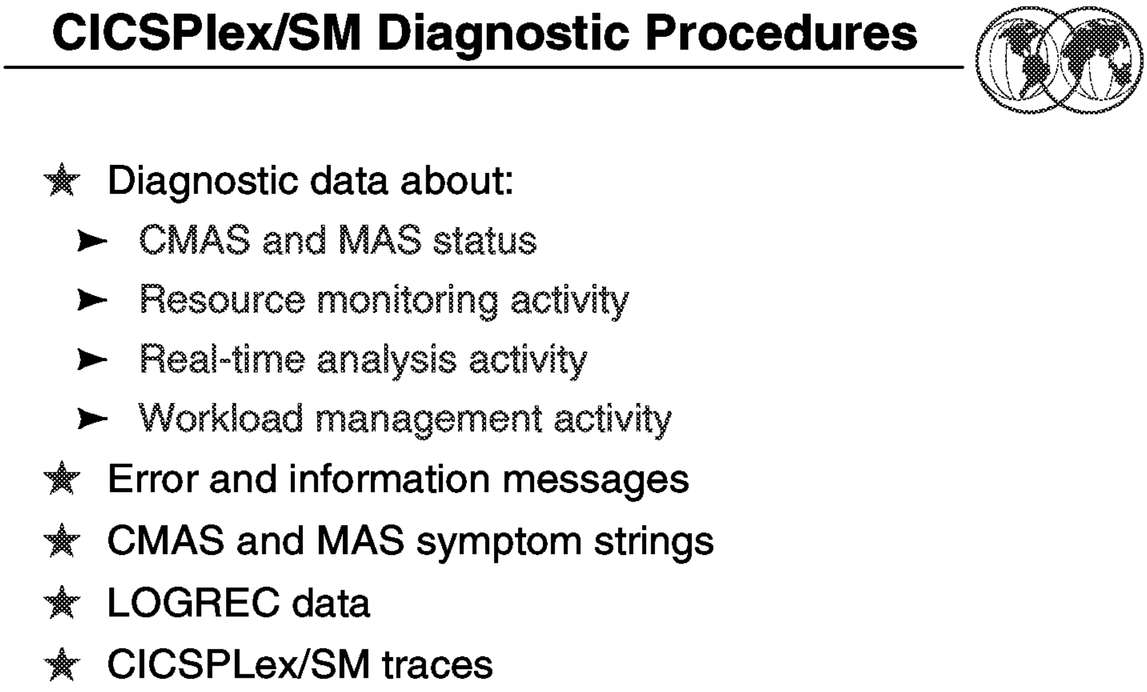
Figure 221 shows the steps needed to find the failing CSECT.

```
12B00B80   47F0F026   01C3C5C5   000000A0   00000148  | .00..CEE.......
4. Take value at offset x′0C′ and add to address 12B00B80+148
2. F CEE PREV
3. Back up 5 bytes this is the beginning of the module/csect/
function 12B00B80:
12B00B90   47F0F001   183F58F0   C31C184E   05EF0000  |.00....0C..+...
12B00BA0   000047F0   303A90E7   D00C58E0   D04C4100  |...0...X}..\}<.
12B00BB0   E0A05500   C3144720   F0145000   E04C9210  |\...C...0.&.\<k
12B00BC0   E00050D0   E00418DE   05304400   C1B04150  |\.&}\.......A..
12B00BD0   00005050   D0984400   C1AC4160   000A8E60  |..&&}q..A..-...
12B00BE0   00205D60   D0985070   D09C4400   C1AC58F0  |..)-}q&.}...A..
1. MCH_PSW points here do L 12B00BE6
12B00BF0   D09C47F0   302E0700   4400C1B8   58D0D004  |}..0......A..}}
12B00C00   58E0D00C   9837D020   051E0707   12B00C20  |.\}.q.}........
12B00C10   12B00CB0   12B00A00   80000001   000006D8  |...............
12B00C20   12B00B80   0000008A   12B00C5C   00000000  |...........*...
12B00C30   02E0004A   00000300   004E0000   03200057  |.\.¢.....+.....
12B00C40   00000360   006B0000   03800078   000003A0  |...-.,.........
12B00C50   00780000   03A0007C   000003A0   0E0E0E0E  |.......@.......
12B00C60   0E0E0000   000F86A4   95836DA6   89A3886D  |......func_with
12B00C70   85999996   99400001   93400001   94400000  |error ..l ..m .
12B00C80   00000692   01012000   00000198   D000009C  |...k.......q}..
12B00C90   0000068E   01012000   00000198   D0000098  |...........q}..
12B00CA0   00000698   000006A8   00000000   00000000  |...q...y.......
12B00CB0   00000000   0000008A   000006B8   00010000  |...............
12B00CC0   00000002   F04A3042   10CEA186   FFFFFB08  |....0¢....f...
5. Go to that location (12B00CC8)
12B00CD0   0000008C   00000000   FFC00000   00000000  |.........{.....
12B00CE0   90000000   02C00019   000F86A4   95836DA6  |....{.....func_w
6. Add X′20′ bytes to get to the 2 byte prefixed string with name
12B00CF0   89A3886D   85999996   99405000   0045FFFF  |ith_error.......
```

Figure 221. Steps required to find the failing CSECT in LE

Some other helpful IPCS LE diagnostic options are:

- VERBX CEEERRIP ′SM′ displays all storage management control blocks.
- VERBX CEEERRIP ′HEAP′ displays HEAP storage management control blocks.
- VERBX CEEERRIP ′STACK′ displays STACK storage management control blocks.
- VERBX CEEERRIP ′ALL′ displays all control blocks.



CICSPlex/SM Diagnostic Procedures

- Diagnostic data about:
  - CMAS and MAS status
  - Resource monitoring activity
  - Real-time analysis activity
  - Workload management activity
- Error and information messages
- CMAS and MAS symptom strings
- LOGREC data
- CICSPLex/SM traces

Figure 222. CICSPlex/SM diagnostic procedures

8.11.3 CICSPlex/SM diagnostic procedures

The diagnostic data that can be collected will not assist you very much in solving application-type problems, as the CICSPlex/SM diagnostic and tracing routines are primarily directed at the "internal" functions of CICSPlex/SM. Despite this, it is extremely important that you understand how to turn on tracing and what dumps are required to help the support center assist with your diagnosis.

There are several online diagnostic aids that can be very useful to prepare diagnostic data.

- CICSPlex SM views that provide diagnostic information about:
  - CMAS and MAS status
  - Resource monitoring activity
  - Real-time analysis activity
  - Workload management activity
- CICS commands that produce data similar to CICSPlex SM data
- The CICSPlex SM online utility transaction (COLU)
- The CICSPlex SM interactive debugging transactions (COD0 and CODB)

Messages are often the first or only indication to a user that something is not working. CICSPlex SM writes error and informational messages to variety of destinations:

- The system console or system log
- The CMAS or MAS job log
- The EYULOG transient data queue
- The SYSOUT data set

- A CICS terminal
- The TSO READY prompt
- The ISPF end-user interface

Any CMAS or local MAS can produce symptom strings in a system or transaction dump. Symptom strings describe a program failure and the environment in which the failure occurred. All CICSPlex SM symptom-strings conform to the RETAIN symptom-string architecture. They are stored as SYMREC records in the SYS1.LOGREC data set.

LOGRECs are records containing information about an abnormal occurrence within CICSPlex SM. The records are written to the SYS1.LOGREC data set and are available for analysis after a failure.

The LOGRECs produced by CICSPlex SM all contain the same data. The data includes extensive information about the state of CICSPlex SM components in the failing address space at the time the LOGREC is written, such as:

- Identification of the failing module
- Module calling sequence
- Recovery management information

8.11.3.1 CICSPlex/SM traces

The CICSPlex SM trace facilities provide a detailed record of every exception condition that occurs. They can also be used to trace various aspects of component processing.

In CMASs and MASs, CICSPlex SM writes user trace records to the CICS trace data set, as follows:

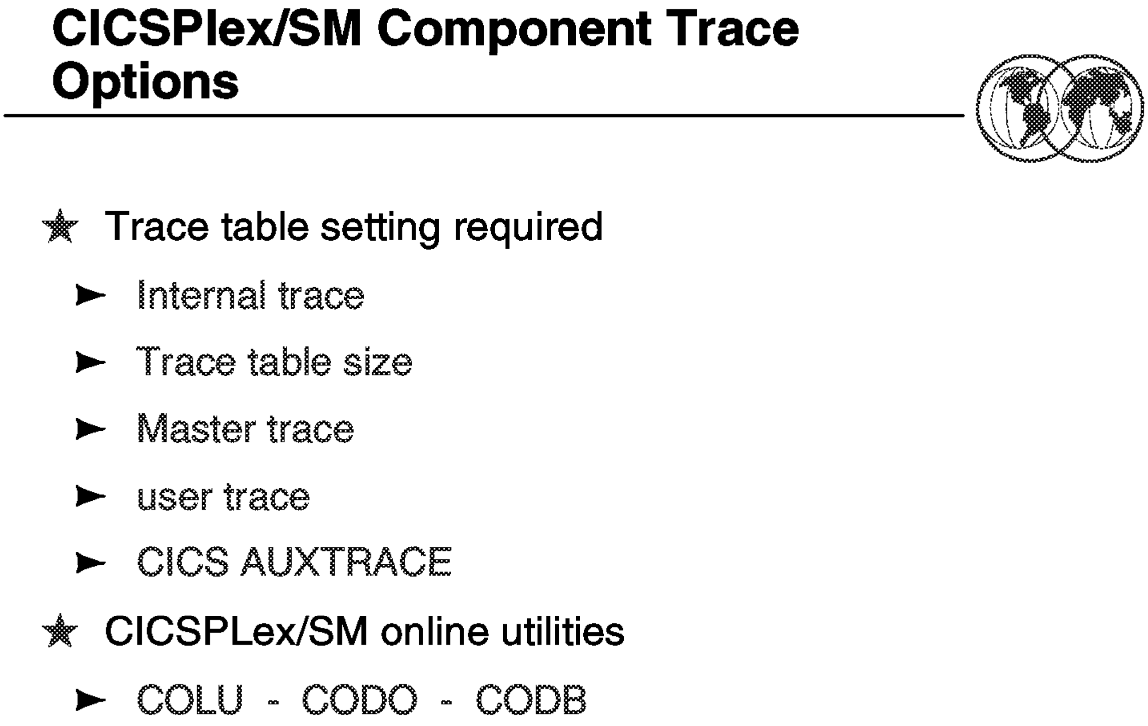
- If any local or remote MAS is in communication with a CMAS, trace data is shipped from the MAS to the CMAS, and a full, formatted trace record is produced.
- If any local or remote MAS is not in communication with a CMAS (either because the communications component is not yet active or because there is a problem with communications itself).

In CASs, trace data is written to a wrap-around buffer, with new data overwriting old data. Because CAS trace data is not written to any external device, it can be examined only within the context of an address space dump.

MVS/ESA system dumps are an important source of detailed information about problems. For CMASs and local MASs, CICSPlex SM recovery routines produce a system dump when an unexpected error occurs in a supervisory function. Users can also request a system dump at any time.

Whether it is the result of an abend or a user request, a system dump provides an exact image of what was happening in a CICSPlex SM address space at the time the dump was taken. A dump can be used to determine the state of all components in the address space, allowing you to:

- Examine MVS/ESA system trace entries
- Determine subsystem status
- Analyze CAS message trace table entries
- Locate key data areas
- Provide information on all CICSPlex SM components
- Provide information on all active CICSPlex SM tasks
- Provide SNAPshots of appropriate CICSPlex SM data spaces



CICSPlex/SM Component Trace Options

★ Trace table setting required

- ► Internal trace
- ► Trace table size
- ► Master trace
- ► user trace
- ► CICS AUXTRACE

★ CICSPLex/SM online utilities

- ► COLU - CODO - CODB

Figure 223. CICSPlex/SM component trace options

8.11.4 CICSPlex/SM component trace options

All CICSPlex SM address space (CMAS), managed application system (MAS), and coordinating address space (CAS) components provide trace data.

Note: The CICS internal trace facilities must always be active in a CMAS.

When a CMAS is initialized, CICSPlex SM ensures that the CICS trace facility is active and the trace table is large enough. The trace table settings required by the CMAS, along with the CICS SIT options that you need to use in order to establish these settings, are:

Internal trace must be *ON*. SIT parameter is INTTR=ON.

Trace table size must be *2MB*. SIT paramater TRTABSZ=2048.

Master trace must be *OFF*. SIT parameter SYSTR=OFF.

User trace must be *ON*. SIT parameter USERTR=ON.

Additionally, the CICS AUXTRACE facility should be active (for user records only) in a CMAS. If this facility is not active when a problem occurs, it may be necessary to recreate the problem with the facility turned on.

During normal CMAS and MAS processing all the standard and special trace levels (levels 1 through 32) are usually disabled. Exception tracing is always active and cannot be disabled.

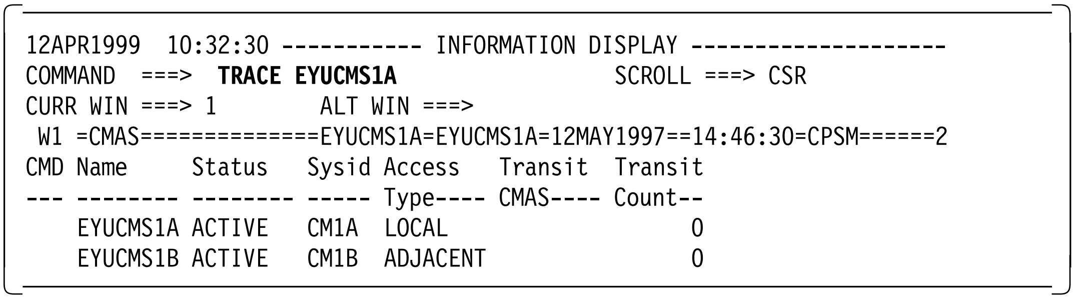
You can turn tracing on for a specific CMAS or MAS component in either of two ways:

- Specify system parameters on a CMAS or MAS startup job.
- Use the ISPF end-user interface to activate one or more levels of tracing dynamically while CICSPlex SM is running.

You can use the CMAS or MAS view to control the tracing that occurs in a active CMAS or MAS.

For example, if you want to change the trace levels for the CMAS called EYUCMS1A, do the following:

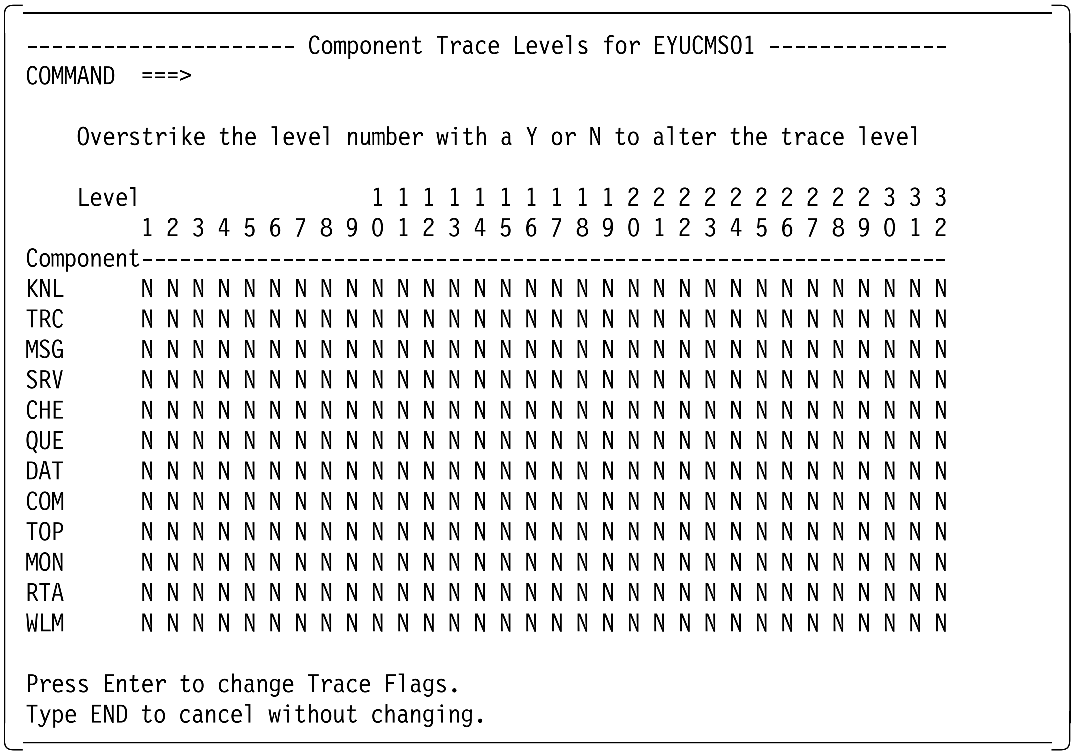
1. Issue the CMAS `VIEW` command.
2. Either issue the `TRACE` primary action command from the COMMAND field, as shown in Figure 224, or enter the `TRA` line action command in the line command field next to EYUCMS1A.



```
12APR1999  10:32:30 ----------- INFORMATION DISPLAY --------------------
COMMAND  ===>  TRACE EYUCMS1A                SCROLL ===> CSR
CURR WIN ===> 1        ALT WIN ===>
 W1 =CMAS=============EYUCMS1A=EYUCMS1A=12MAY1997==14:46:30=CPSM======2
CMD Name     Status   Sysid Access   Transit  Transit
--- -------- -------- ----- Type---- CMAS---- Count--
    EYUCMS1A ACTIVE   CM1A  LOCAL                   0
    EYUCMS1B ACTIVE   CM1B  ADJACENT                0
```

Figure 224. Contolling CICSplex/SM tracing from a CMAS view

3. The component trace input panel appears, as shown in Figure 225, which identifies the current trace settings for each component in the CMAS. A setting of *Y* means that trace level is active for the component; a setting of *N* means tracing is not active.



```
--------------------- Component Trace Levels for EYUCMS01 --------------
COMMAND  ===>

    Overstrike the level number with a Y or N to alter the trace level

    Level                  1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 3 3 3
         1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2
Component---------------------------------------------------------------
KNL      N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
TRC      N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
MSG      N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
SRV      N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
CHE      N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
QUE      N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
DAT      N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
COM      N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
TOP      N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
MON      N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
RTA      N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
WLM      N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N

Press Enter to change Trace Flags.
Type END to cancel without changing.
```

Figure 225. CICSPlex/SM component trace options

Note

Level 3 through 32 trace points should be activated only for a specific CMAS or MAS component and only at the request of customer support personnel.

4. To change a trace setting for a specific component, such as Kernel Linkage (KNL):

   a. Position the cursor next to KNL.

   b. Move the cursor across the line to the appropriate level (1 through 32).

   c. Type either Y, to activate tracing, or N, to deactivate tracing.

5. When all the trace settings are correct, press Enter. The CMAS view is redisplayed.

CICSPlex/SM provides three online transactions. These are:

- COLU
- COD0
- CODB

The CICSPlex SM online utility (COLU) is a CICS transaction that can be used to generate reports about various CMAS and local MAS components. The interactive debugging transactions COD0 and CODB provide access to the CICSPlex SM run-time environment. They can be used to format and manipulate the internal data structures of CICSPlex SM. The debugging transactions can run in CMASs, and in CICS/ESA, CICS/MVS, and CICS/VSE MASs with terminal support. Transaction COD0, with some variations, is also supported in CICS for OS/2 MASs. Transaction CODB is not supported in CICS for OS/2 MASs.

Attention

The CICSPlex SM interactive debugging transactions COD0 and CODB and online utility COLU should be used only at the request of IBM customer support personnel. You must take steps to ensure that these transactions may be used only by authorized personnel because of the extent of the access to system control areas that they provide. Improper or unauthorized use of COD0 and CODB may have very serious consequences, including without limitation loss of data or system outage. The customer shall be solely responsible for such misuse.

CICSPlex/SM Dumps

- Use MVS DUMP command
- SNAP command - CICSRGN view
- SNAP Input Panel
- Dump Analysis Panel
- CICSPLex/SM Subsystem Component Analysis Panel

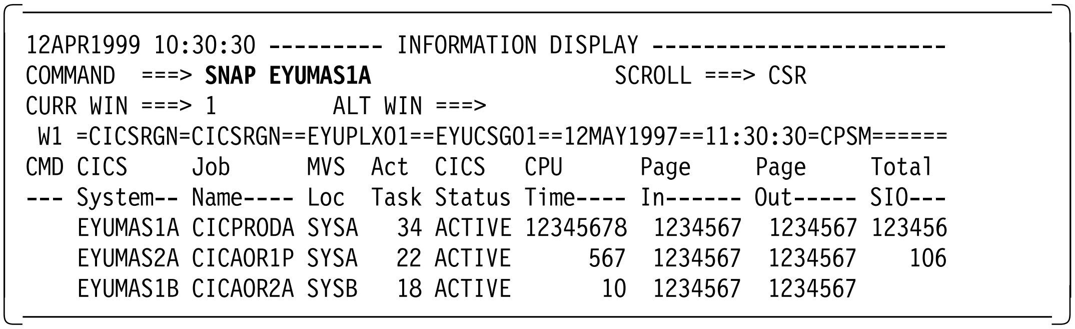
Figure 226. CICSPlex/SM dumps

8.11.5 CICSPlex/SM dumps

A dump of a CICSPlex SM address space can be initiated by the MVS DUMP command or via the CICSPlex/SM ISPF end-user interface. You can use the SNAP action command on the CICSRGN view to request a system dump for an MAS.

For example, if you want a system dump for the MAS called EYUMAS1A, do following:

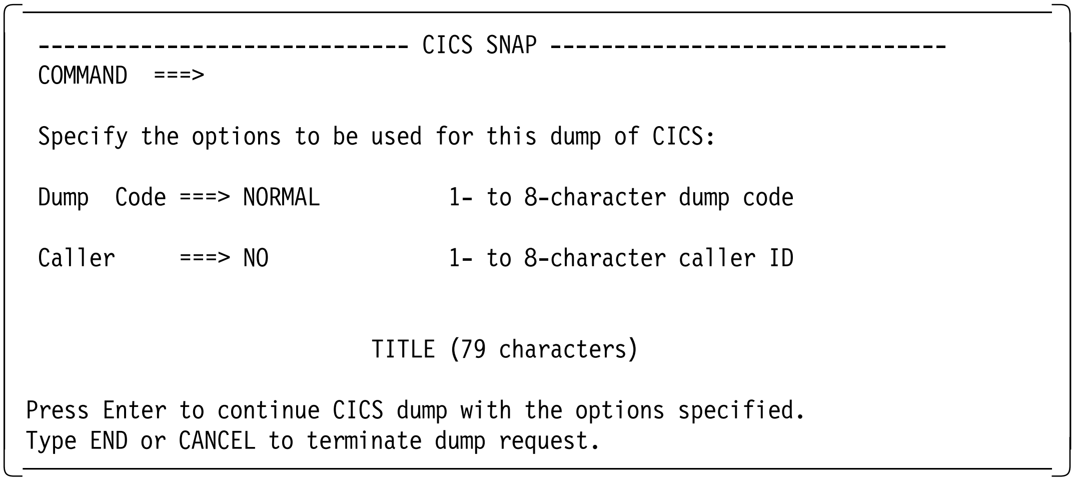
1. Issue the CICSRGN view command.
2. Either issue the SNAP primary action command from the COMMAND field, as shown in Figure 227 or enter the SNA line action command in the line command field next to EYUMAS1A.



```
12APR1999 10:30:30 --------- INFORMATION DISPLAY -----------------------
COMMAND  ===> SNAP EYUMAS1A                  SCROLL ===> CSR
CURR WIN ===> 1         ALT WIN ===>
 W1 =CICSRGN=CICSRGN==EYUPLX01==EYUCSG01==12MAY1997==11:30:30=CPSM======
CMD CICS     Job      MVS  Act  CICS   CPU      Page     Page     Total
--- System-- Name---- Loc  Task Status Time---- In------ Out----- SIO---
    EYUMAS1A CICPRODA SYSA   34 ACTIVE 12345678  1234567  1234567 123456
    EYUMAS2A CICAOR1P SYSA   22 ACTIVE      567  1234567  1234567    106
    EYUMAS1B CICAOR2A SYSB   18 ACTIVE       10  1234567  1234567
```

Figure 227. Initiate CICSPlex/SM system dump Via SNAP

3. When the CICS SNAP input panel appears, as shown in Figure 227 on page 331 specify:
   - A 1- to 8-character dump code
   - An optional 1- to 8-character caller ID
   - An optional title of up to 79 characters



```
 ---------------------------- CICS SNAP ------------------------------
 COMMAND  ===>

 Specify the options to be used for this dump of CICS:

 Dump  Code ===> NORMAL         1- to 8-character dump code

 Caller     ===> NO             1- to 8-character caller ID


                          TITLE (79 characters)

Press Enter to continue CICS dump with the options specified.
Type END or CANCEL to terminate dump request.
```

Figure 228. CICSPlex/SM system dump via SNAP input panel

The following message appears in the window to confirm the dump request:

```
EYUEI0568I  Dump Taken for EYUMAS1A, assigned DUMPID is nn/nnnn
```

Where nn/nnnn is the dump ID assigned by MVS.

In the job log for EYUMAS1A you will see:

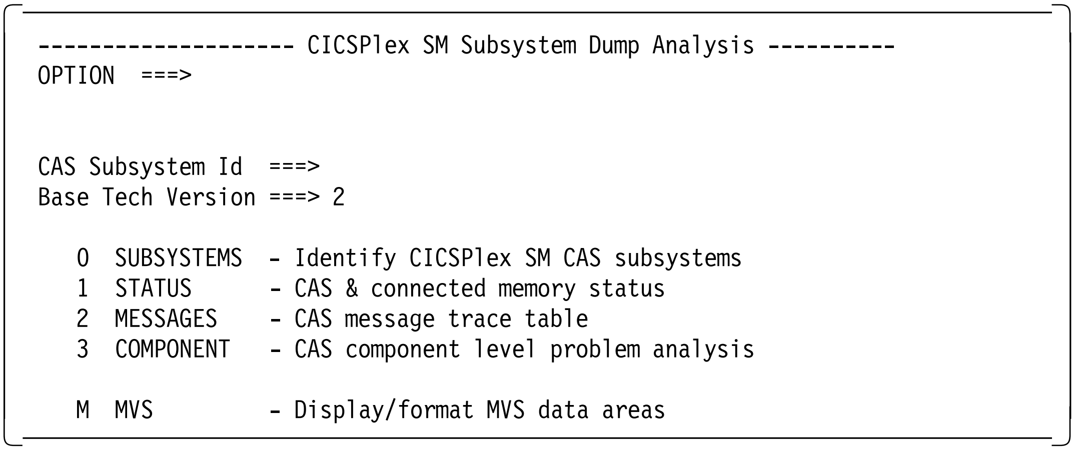
```
10.03.05 JOB00221 +DFHDU0201 EYUMAS1A ABOUT TO TAKE SDUMP. DUMPCODE:code
10.03.12 JOB00221 IEA794I SVC DUMP HAS CAPTURED:
                  DUMPID=005 REQUESTED BY JOB (EYUMAS1A)
                  DUMP TITLE=CICSDUMP: SYSTEM=EYUMAS1A CODE=code ID=nn
10.03.05 JOB00221 +DFHDU0202 EYUMAS1A SDUMP COMPLETE.
```

The interactive problem control system (IPCS) provides MVS users with an interactive facility for diagnosing software failures. You can use IPCS to format and analyze SDUMPs produced by CICSPlex SM or stand-alone dump obtained while CICSPlex SM was active in the system being dumped. You can either view the dumps at your terminal or print them.

CICSPlex SM provides two types of IPCS tools:

- A set of panels (driven by a corresponding set of CLISTs) that allow you to display:
  – The data in a coordinating address space (CAS) dump
  – The names and locations of control blocks and areas of a CAS dump
  – Subsystem information
  – Address space-related control blocks
  – Modules loaded by CICSPlex SM
  – Tasks created by CICSPlex SM
  – Storage subpools managed by CICSPlex SM
  – BBC LU 6.2 communication information

- A dump formatting routine that can be used with the VERBEXIT subcommand to format CMAS or MAS dumps



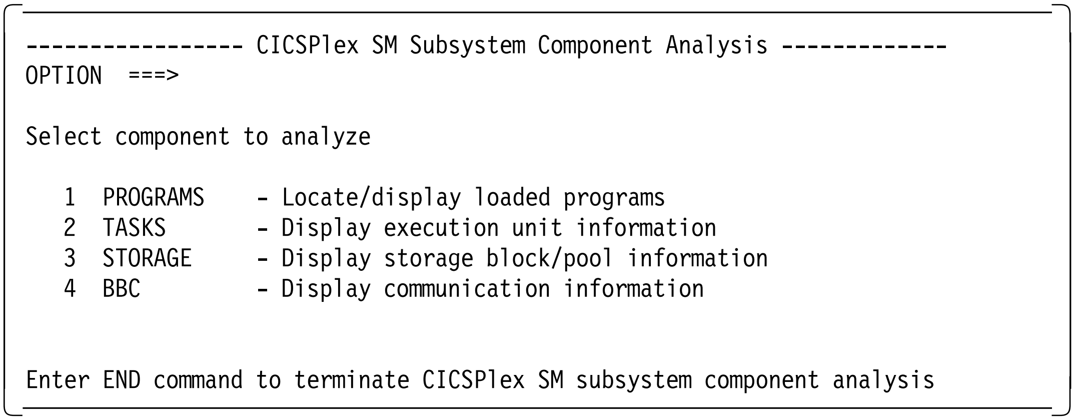
```
-------------------- CICSPlex SM Subsystem Dump Analysis ----------
OPTION  ===>


CAS Subsystem Id  ===>
Base Tech Version ===> 2

   0  SUBSYSTEMS  - Identify CICSPlex SM CAS subsystems
   1  STATUS      - CAS & connected memory status
   2  MESSAGES    - CAS message trace table
   3  COMPONENT   - CAS component level problem analysis

   M  MVS         - Display/format MVS data areas
```

Figure 229. CICSPlex/SM Subsystem Dump Analysis panel

Figure 229 shows the subsystem dump analysis panel that can be invoked from:

- The IPCS primary option menu, select option (2) ANALYSIS
- The IPCS MVS analysis menu, select option (6) COMPONENT
- The IPCS MVS component menu, select CPSMSSDA

After identifying the CAS subsystem, selecting **option 3**, COMPONENTS, will display the panel shown in Figure 230.



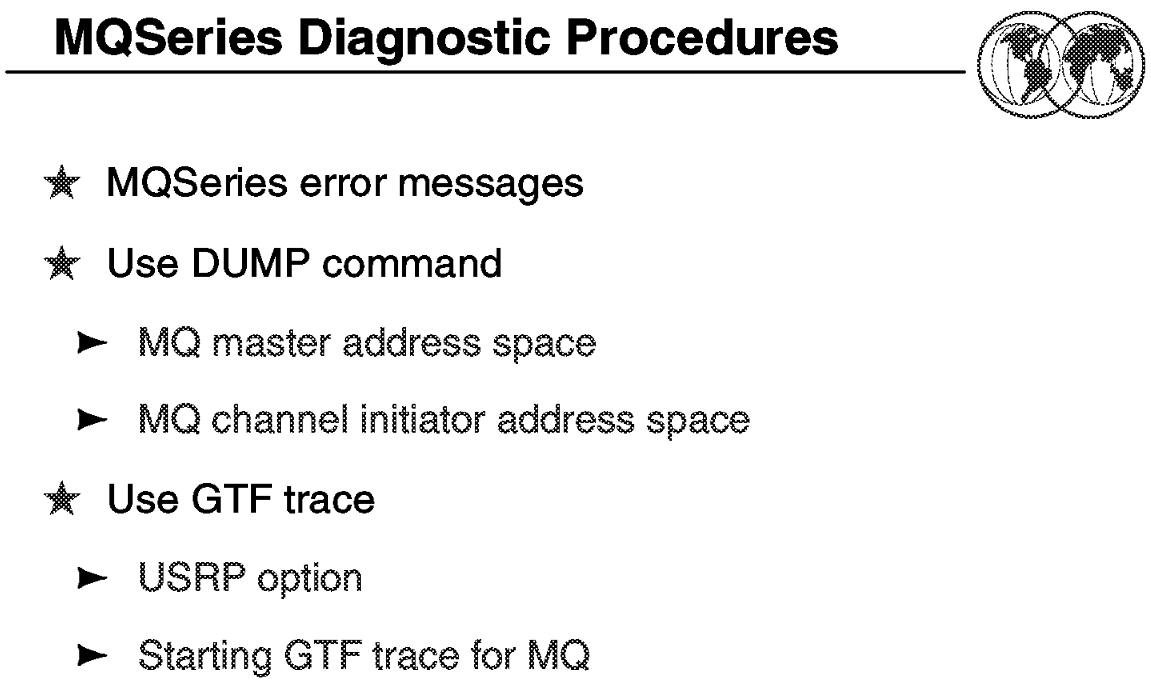
```
----------------- CICSPlex SM Subsystem Component Analysis -------------
OPTION  ===>

Select component to analyze

   1  PROGRAMS    - Locate/display loaded programs
   2  TASKS       - Display execution unit information
   3  STORAGE     - Display storage block/pool information
   4  BBC         - Display communication information


Enter END command to terminate CICSPlex SM subsystem component analysis
```

Figure 230. CICSPlex/SM component analysis

CICSPlex/SM trace data can also be formatted using IPCS VERBX command:

```
VERBX EYU9Dxxx 'TRC=1'
```

Where xxx is the CICSPlex/SM release.



MQSeries Diagnostic Procedures

- MQSeries error messages
- Use DUMP command
  - MQ master address space
  - MQ channel initiator address space
- Use GTF trace
  - USRP option
  - Starting GTF trace for MQ

Figure 231. MQSeries diagnostic procedures

8.12 MQSeries diagnostic procedures

MQSeries for MVS/ESA provides unique messages that, together with the output of dumps, are aimed at providing sufficient data to allow diagnosis of the problem without having to try to reproduce it.

MQSeries for MVS/ESA tries to produce an error message when a problem is detected. MQSeries for MVS/ESA diagnostic messages all begin with the prefix CSQ. Each error message generated by MQSeries is unique; it is generated for one and only one error. Information about the error can be found in *MQSeries for MVS/ESA Messages and Codes*, GC33-0819. The first three characters of the names of MQSeries for MVS/ESA modules are also CSQ. The fourth character uniquely identifies the component. Characters five through eight are unique within the group identified by the first four characters. There may be some instances when no message is produced, or, if one is produced, it cannot be communicated. In these circumstances, you might have to analyze a dump to isolate the error to a particular module.

When capturing diagnostic data to analyze MQ problems ensure that you dump both the MQ main (MSTR) and channel initiator (CHIN) address spaces. Figure 232 on page 335 shows the procedure to dump the MQ MSTR address space.

```
DUMP COMM=(MQSERIES MAIN DUMP)
*01 IEE094D SPECIFY OPERAND(S) FOR DUMP COMMAND
R 01,JOBNAME=CSQ1MSTR,CONT
*02 IEE094D SPECIFY OPERAND(S) FOR DUMP COMMAND
IEE600I REPLY TO 01 IS;JOBNAME=CSQ1MSTR,CONT
R 02,SDATA=(CSA,RGN,PSA,SQA,TRT),END
IEE600I REPLY TO 02 IS;SDATA=(CSA,RGN,PSA,SQA,TRT),END
IEA794I SVC DUMP HAS CAPTURED: 869
DUMPID=001 REQUESTED BY JOB (*MASTER*)
DUMP TITLE=MQSERIES MAIN DUMP
```

Figure 232. Procedure to DUMP the MQ master address space

Figure 233 shows the procedure to dump the MQ CHIN address space.

```
DUMP COMM=(MQSERIES CHIN DUMP)
*01 IEE094D SPECIFY OPERAND(S) FOR DUMP COMMAND
R 01,JOBNAME=CSQ1CHIN,CONT
*02 IEE094D SPECIFY OPERAND(S) FOR DUMP COMMAND
IEE600I REPLY TO 01 IS;JOBNAME=CSQ1CHIN,CONT
R 02,SDATA=(CSA,RGN,PSA,SQA,TRT),CONT
*03 IEE094D SPECIFY OPERAND(S) FOR DUMP COMMAND
IEE600I REPLY TO 02 IS;SDATA=(CSA,RGN,PSA,SQA,TRT),CONT
R 03,DSPNAME=′CSQ1CHIN′.CSQXTRDS,END
IEE600I REPLY TO 03 IS;DSPNAME=′CSQ1CHIN′.CSQXTRDS,END
IEA794I SVC DUMP HAS CAPTURED: 869
DUMPID=001 REQUESTED BY JOB (*MASTER*)
DUMP TITLE=MQSERIES CHIN DUMP
```

Figure 233. Procedure to DUMP the MQ channel initiator address space

You can obtain information about API calls and user parameters passed by some MQSeries calls on entry to, and exit from, MQSeries. To do this, you should use the global trace in conjunction with the MVS generalized trace facility (GTF). To use the trace for problem determination, you must start the following

- The GTF for your MVS system
- The MQSeries trace for each queue manager subsystem for which you want to collect data

When you start the GTF, you should specify the USRP option. You will be prompted to enter a list of event identifiers (EIDs). The EIDs used by MQSeries are:

- *5E9* To collect information about control blocks on entry to MQSeries
- *5EA* To collect information about control blocks on exit from MQSeries
- *5EB* To collect information on entry to internal MQ functions
- *5EE* To collect information on exit from internal MQ functions

You can also use the JOBNAMEP option, specifying the batch, CICS, IMS, or TSO job name, to limit the trace output to certain jobs.

```
 START GTFxx.xx
  HASP100 GTFxx.xx  ON STCINRDR
  HASP373 GTFxx.xx  STARTED
*01 AHL100A SPECIFY TRACE OPTIONS
 R 01,TRACE=JOBNAMEP,USRP
 TRACE=JOBNAMEP,USRP
 IEE600I REPLY TO 12 IS;TRACE=JOBNAMEP,USRP
*02 ALH101A SPECIFY TRACE EVENT KEYWORDS - JOBNAME=,USR=
 R 02,JOBNAME=jobname,USR=(5E9,5EA,5EB,5EE)
 JOBNAME=jobname,USR=(5E9,5EA,5EB,5EE)
 IEE600I REPLY TO 13 IS;JOBNAME=jobname,USR=(5E9,5EA,5EB,5EE)
*03 ALH102A CONTINUE TRACE DEFINITION OR REPLY END
 R 03,END
 END
 IEE600I REPLY TO 14 IS;END
 AHL103I TRACE OPTIONS SELECTED-USR=(5EA,5E9,5EB,5EE)
 AHL103I JOBNAME=(jobname)
*04 AHL125A RESPECIFY TRACE OPTIONS OR REPLY U
 R 04,U
 U
 IEE600I REPLY TO 15 IS;U
 AHL031I GTF INITIALIZATION COMPLETE
```

Figure 234. Starting a GTF trace for MQ

Figure 234 shows how to start a GTF trace for MQ.

You must then START TRACE in the MQ environment and this data will be written to GTF. The format of this command is:

```
smqid START TRACE
```

Where s is the subsystem identifier for MQ and mqid is the MQ subsystem name that you want to trace. The MQ subsystem identifier and name are defined in member IEFSSNxx in SYS1.PARMLIB. Use the START TRACE command, specifying type GLOBAL to start writing MQSeries records to the GTF. To define the events that you want to produce trace data for, use one or more of the following classes:

CLASS	Event traced
2	Record the API call and API parameters when a completion code other than MQRC_NONE is detected.
3	Record the API call and API parameters on entry to and exit from the queue manager.

Once started, you can display information about, and alter the properties of and stop, the trace with the DISPLAY TRACE, ALTER TRACE, and STOP TRACE commands.

To use any of the trace commands, you must have one of the following:

- Authority to issue start/stop trace commands (trace authority)
- Authority to issue the display trace command (display authority)

To format the user parameter data collected by the global trace you can use the IPCS GTFTRACE USR(xxx) command, where xxx is:

5E9 To format information about control blocks on entry to MQSeries MQI calls

5EA To format information about control blocks on exit from MQSeries MQI calls

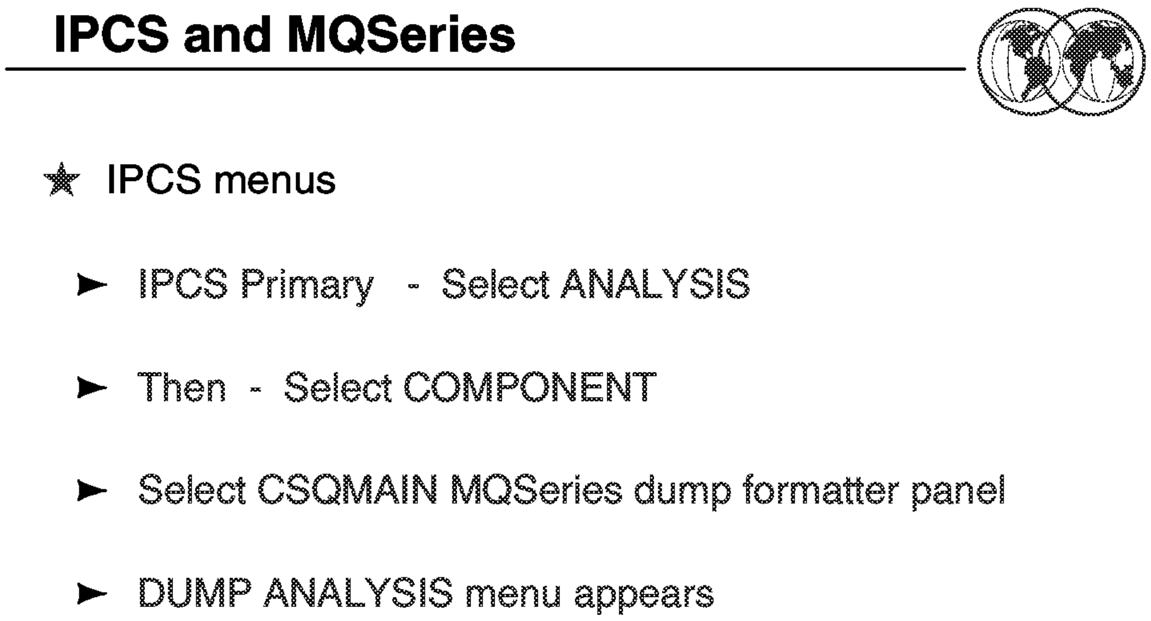
5E9,5EA To format information about control blocks on entry to and exit from MQSeries MQI calls

5EB,5EE To format information about control blocks on entry to and exit from MQSeries Internal functions

You can also specify the JOBNAME(jobname) parameter, to limit the formatted output to certain jobs. Figure 235 shows a sample MQ 5EA trace entry.

```
USRD9 5EA ASCB 00F31200           JOBN MQ69CHIN
CSQW073I EXIT:  MQSeries user parameter trace
OPEN
    Thread... 008D7838  Userid... COAKLEY   pObjDesc. 0EBB2F38
    RSV1..... 00000000  RSV2..... 00000000  RSV3..... 00000000
    CompCode. 00000000  Reason... 00000000
    D4D8F6F9  C3C8C9D5  00000000  00000000  | MQ69CHIN........ |
    E3F0F0F8  C4F7F8F3  F8404040  40404040  | T008D7838        |
    40404040  40404040  40404040  40404040  |                  |
    40404040  40404040  40404040  40404040  |                  |

USRD9 5EA ASCB 00F31200           JOBN MQ69CHIN
CSQW073I EXIT:  MQSeries user parameter trace
 +0000  D6C44040  00000001  00000005  00000000  | OD  ............ |
 +0010  00000000  00000000  00000000  00000000  | ................ |
 +0020  00000000  00000000  00000000  00000000  | ................ |
 +0030  00000000  00000000  00000000  D4D8F6F9  | ............MQ69 |
 +0040  40404040  40404040  40404040  40404040  |                  |
 +0050  40404040  40404040  40404040  40404040  |                  |
 +0060  40404040  40404040  40404040  C3E2D84B  |             CSQ. |
 +0070  5C000000  00000000  00000000  00000000  | *............... |
 +0080  00000000  00000000  00000000  00000000  | ................ |
 +0090  00000000  00000000  00000000  00000000  | ................ |
 +00A0  00000000  00000000                      | ........         |
```

Figure 235. MQ 5EA trace entry



IPCS and MQSeries

★ IPCS menus

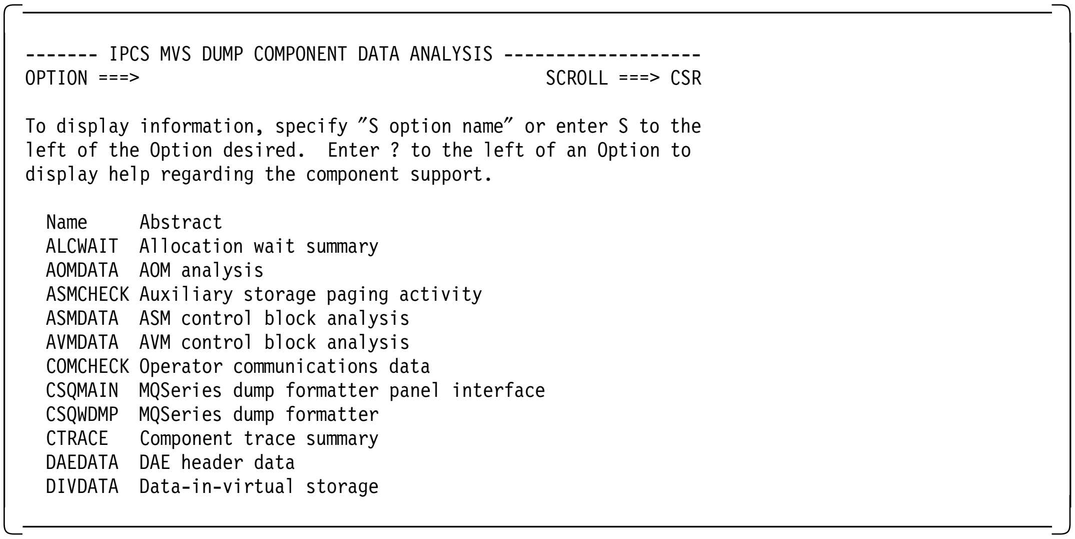
- ➤ IPCS Primary - Select ANALYSIS
- ➤ Then - Select COMPONENT
- ➤ Select CSQMAIN MQSeries dump formatter panel
- ➤ DUMP ANALYSIS menu appears

Figure 236. IPCS and MQSeries

8.12.1 IPCS and MQSeries

MQSeries for MVS/ESA provides a set of panels to help you process dumps using IPCS.

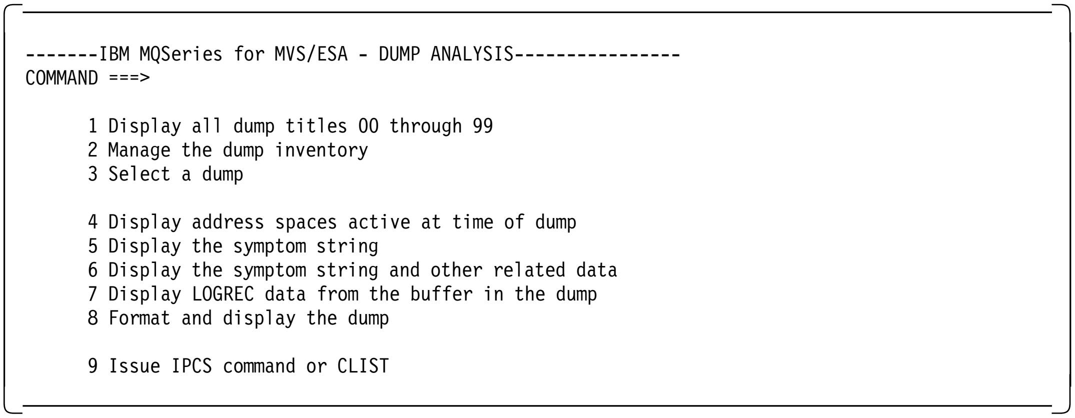
1. From the IPCS Primary Option menu select ANALYSIS (Analyze dump contents). The IPCS MVS Analysis Of Dump Contents panel appears.
2. Select COMPONENT - MVS component data (Option 6). The IPCS MVS Dump Component Data Analysis panel appears. The appearance depends on the products installed at your installation, but will be similar to that shown in Figure 237 on page 339.



```
------- IPCS MVS DUMP COMPONENT DATA ANALYSIS -------------------
OPTION ===>                                       SCROLL ===> CSR

To display information, specify ″S option name″ or enter S to the
left of the Option desired.  Enter ? to the left of an Option to
display help regarding the component support.

  Name     Abstract
  ALCWAIT  Allocation wait summary
  AOMDATA  AOM analysis
  ASMCHECK Auxiliary storage paging activity
  ASMDATA  ASM control block analysis
  AVMDATA  AVM control block analysis
  COMCHECK Operator communications data
  CSQMAIN  MQSeries dump formatter panel interface
  CSQWDMP  MQSeries dump formatter
  CTRACE   Component trace summary
  DAEDATA  DAE header data
  DIVDATA  Data-in-virtual storage
```

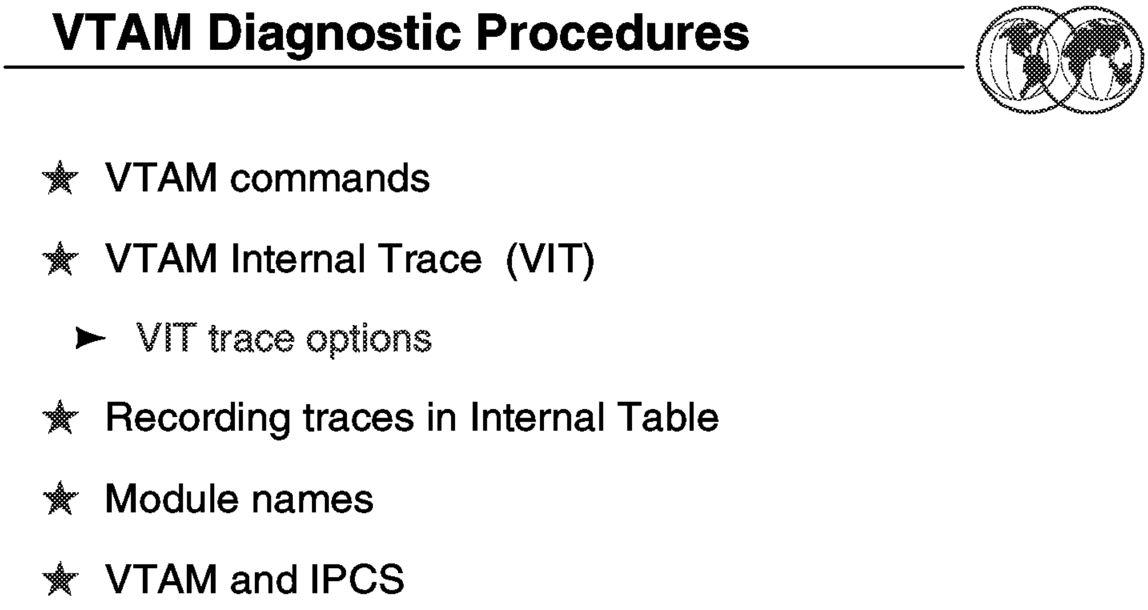
Figure 237. IPCS MVS Dump Component Data Analysis panel

3. Select CSQMAIN MQSeries dump formatter panel interface selecting this item.
4. The IBM MQSeries for MVS/ESA - Dump Analysis menu appears as shown in Figure 238.



```
-------IBM MQSeries for MVS/ESA - DUMP ANALYSIS----------------
COMMAND ===>

      1 Display all dump titles 00 through 99
      2 Manage the dump inventory
      3 Select a dump

      4 Display address spaces active at time of dump
      5 Display the symptom string
      6 Display the symptom string and other related data
      7 Display LOGREC data from the buffer in the dump
      8 Format and display the dump

      9 Issue IPCS command or CLIST
```

Figure 238. IPCS MQSeries for MVS/ESA Dump Analysis panel



VTAM Diagnostic Procedures

- VTAM commands
- VTAM Internal Trace (VIT)
  - VIT trace options
- Recording traces in Internal Table
- Module names
- VTAM and IPCS

Figure 239. VTAM diagnostic procedures

8.13 VTAM diagnostic procedures

The solution to many VTAM problems can be identified by issuing some relevant VTAM DISPLAY commands that can identify problems for devices, lines, major nodes, cross domain resources and buffers, but this is not the exclusive list.

Some useful commands are:

- D NET,PENDING, which displays resources that are in a pending status. This could indicate the resource that is causing a hang during processing or shutdown.
- D NET,CDRMS displays the cross-domain resources.
- D NET,ID=name displays the status of the named device.
- D NET,BFRUSE displays the VTAM buffer statistics.
- V NET,ACT,ID=name, will ACTivate the named resource.
- V NET,INACT,ID=name will INACTivate the named resource.

Messages generated by VTAM and more importantly the sense data that accompanies these messages can be invaluable when diagnosing problems. Figure 240 on page 341 shows a sample VTAM error message.

```
IST663I IPS SRQ REQUEST TO ISTAPNCP FAILED, SENSE=08570003 621
IST664I REAL  OLU=USIBMSC.SC55ALUC    REAL  DLU=USIBMSC.SC54ALUC
IST889I SID = ED03979A6B8E83B5
IST264I REQUIRED RESOURCE SC54ALUC NOT ACTIVE
IST891I USIBMSC.SC54M GENERATED FAILURE NOTIFICATION
IST314I END
```

Figure 240. VTAM error message

The error condition identified in Figure 240 is obvious, and *ACT*ivating resource SC54ALUC would be a good place to start. If activating this device was unsuccessful, then the sense data provided could be used to identify the cause.

Alternatively, the more complex VTAM problems can require a dump of the VTAM address, and/or a dump of the NCP, and a VTAM internal trace.

A dump of the NCP should be taken whenever the NCP abnormally terminates or when an error is suspected in the NCP. It may be possible to determine that a problem exists in the NCP by using the VTAM I/O trace to determine what PIUs are being sent to and received from the communication control and by using the NCP line trace to determine what is happening on the lines between the communication controller and the link-attached logical unit.

Dumping the Network Control Program (NCP) can be started as follows:

```
F vtam_procname,DUMP,ID=ncp_name
```

8.13.1.1 VTAM internal trace (VIT)

Most VTAM traces show the information flow between the VTAM program and other network components. However, the VTAM internal trace (VIT) provides a record of the sequence of events within VTAM. These internal events include the scheduling of processes (for example, POST, WAIT, and DISPATCH), the management of storage (for example, VTALLOC), and the flow of internal PIUs between VTAM components. Together with the operator console listing and a dump, output from the VIT can help you reconstruct sequences of VTAM events and find internal VTAM problems more easily.

Trace data for the following VIT options is always automatically recorded in the internal table:

- API
- MSG
- NRM
- PIU
- SSCP

Use one of the following methods to start the VIT:

- You can use the TRACE start option, with TYPE=VTAM specified, to start the VIT when you first start VTAM.
- You can use the MODIFY TRACE command, with TYPE=VTAM specified, to start the VIT after you have started VTAM.

To prevent the VIT table from being overwritten, VTAM disables the internal VIT when it issues SDUMP and when an FFST probe is tripped. The minimum trace table size is 50 pages, and because the five trace option defaults are always running, the table may wrap many times.

Both the TRACE start option and the MODIFY TRACE command have an OPTION operand you can use to select VIT options. For a description of the VIT options, see D.2, “VTAM internal trace options” on page 380.

VTAM can write the VIT trace data to an internal table or an external device, such as a disk or tape. You specify internal or external with MODE operand of the TRACE start option or the MODIFY TRACE command. The VIT record contains the same information regardless of the mode selected.

You can record data externally and internally at the same time, and if desired, you can have different sets of trace options active for each mode. The default trace options API, MSG, NRM, PIU, and SSCP are always recorded internally.

8.13.1.2 Recording traces in internal table (MODE=INT)

If you set MODE=INT on the MODIFY TRACE command or as a TRACE start option, or if you let MODE default to INT, VTAM writes the VIT trace records in an internal trace table. The table is allocated and initialized in extended common service area (CSA) storage.

The SIZE operand of the TRACE start option specifies the number of pages (1 through 999) in storage to be allocated for the internal trace table. Each page is 4 KB. If you omit this option, the default size is 100 pages. If you specify fewer than 100 pages, VTAM uses 100. Because it is a wraparound table, specify enough pages to ensure that the VIT will not overwrite important trace records when the table fills and begins to wrap around.

8.13.1.3 Recording traces in external file (MODE=EXT)

When you specify MODE=EXT, information is still written to the internal trace table for the default options. The external trace file is produced by GTF, and the default file name is SYS1.TRACE. You can print the internal trace data with IPCS or TAP. If you use IPCS to print the data, specify the GTFTRACE option, and set USR(FE1).

8.13.1.4 Module Names in the internal trace records

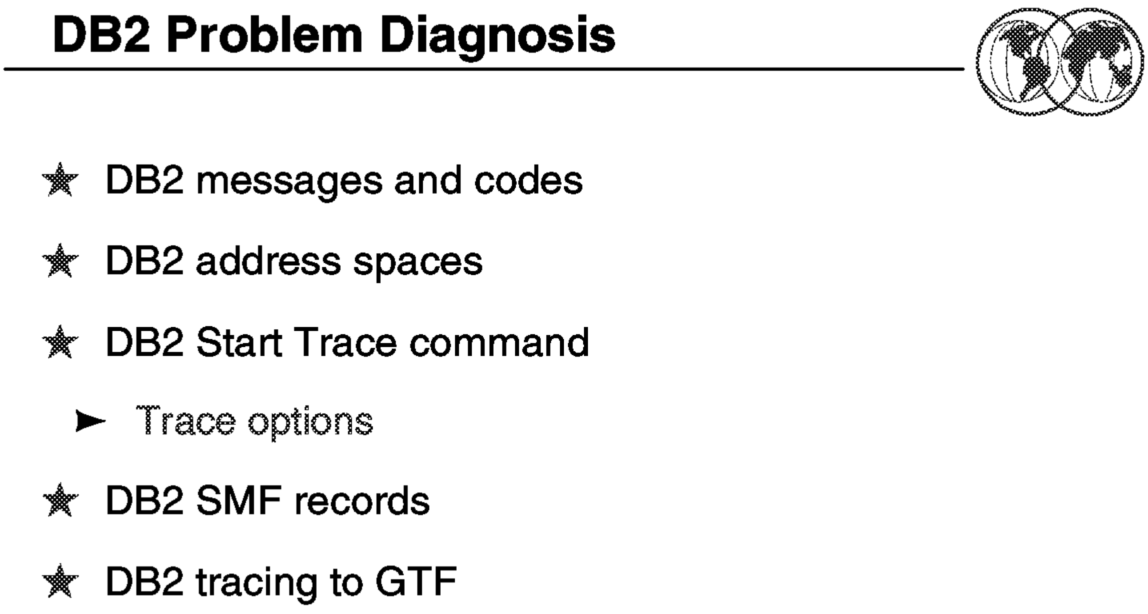
Many VTAM internal trace records include the associated module names in EBCDIC, without the first prefix and, for some types of trace records, without the sixth letter. For example, you would see TSSR for module ISTTSCSR. You can save time by scanning for these module names when you are following the logic flow through VTAM. You can sometimes isolate a VTAM problem to a specific component or module without even looking at a dump. Module names can also be determined from the ISSR field in some VIT records. If the issuer is an LPA module , the address can be found in the VTAM module list (which currently contains LPA modules).

8.13.1.5 VTAM and IPCS

The IPCS subcommand VERBEXIT VTAMMAP can be used to format VTAM dumps. This will format the VTAM internal trace table as well as VTAM control blocks. Figure 241 on page 343 shows sample VIT entries from the VERBEXIT VTAMMAP command.

```
                            ALL Analysis

VTAM INTERNAL TRACE TABLE    0A4E6000

PRESENT WRAP  B221816393ADCA02  LAST WRAP    B22181376BD8F400

CURRENT ENTRY 0A52DCE0          LAST ENTRY   0A549FE0
SCHD        ASID 00                  FLGS  280012   PST   0A740360 PAB
                                                    WEQ   00000000 NAME
EXIT        ASID 1A   APNOP 00       PABOF 04C8     PST   0A740360 PAB
                                                    WEQ   80000000 NAME
DSP         ASID 00   CBID  00       FLAG  80       FLAG1 12       LEVEL
                                                    LAST  00000000 WEA
PIU         ASID 1A   CBID  99       FLAG  00       TSCB  0A6537A8
                                                    TH    40000000 00000
PIU2        PIU  011902AC 00110390 207DC77C 11C7F497 99858689 A7405C00 0
QUE         ASID 1A   CBID  99       FLGS  400110   PST   0A6D6978 PAB
                                                    WEA   0A6537A8 NAME
REQS        ASID 1A   CBID  01       PST   0A740360 BUF   0A6A6870 ISSR
```

Figure 241. VTAM internal trace - VERBEXIT VTAMMAP



DB2 Problem Diagnosis

- DB2 messages and codes
- DB2 address spaces
- DB2 Start Trace command
  - Trace options
- DB2 SMF records
- DB2 tracing to GTF

Figure 242. DB2 problem diagnosis

8.14 DB2 problem diagnosis

As with all problems, the messages and codes generated by the DB2 subsystem should be reviewed for symptoms. When diagnosing DB2 problems it will be necessary to dump the DB2 address spaces. DB2 consists of more than one address space. Depending on your installation these could be:

- DB2 master (MSTR) address space
- Data base manager (DBM) address space
- Internal resource lock manager (IRLM) address space
- Distributed data facility (DIST) address space
- Established stored procedures (SPAS) address space

Usually you would dump the MSTR, DBM, and IRLM address spaces.

The DB2 command START TRACE starts DB2 traces. This command can be issued from an MVS console, a DSN session, a DB2I panel (DB2 COMMANDS), an IMS or CICS terminal, or a program using the instrumentation facility interface (IFI). To execute this command, the privilege set of the process must include one of the following:

- TRACE privilege
- SYSOPR, SYSCTRL, or SYSADM authority

DB2 commands issued from an MVS console are not associated with any secondary authorization IDs. The format of the START TRACE command is:

```
START TRACE (PERFM)   DEST(GTF)   PLAN(plan_name,..) CLASS(class)
        (ACCTG)       (SMF)
        (STAT)        (SRV)
        (AUDIT)       (OP)
        (MONITOR)     (OPX)
```

You must specify a trace type. The options PERFM, ACCTG, STAT, AUDIT, and MONITOR identify the type of trace started, as follows:

(PERFM) Is intended for performance analysis and tuning, and includes a record of specific events in the system.

(ACCTG) Is intended to be used in accounting for a particular program or authorization ID, and includes records written for each thread.

(STAT) Collects statistical data broadcast by various components of DB2, at time intervals that can be chosen during installation.

(AUDIT) Collects audit data from various components of DB2.

(MONITOR) Collects monitor data. Makes trace data available to DB2 monitor application programs.

The DEST option specifies where the trace output is to be recorded. You can use more than one value, but do not use the same value twice. If you do not specify a value, the trace output is sent to the default destination. If the specified destination is not active or becomes inactive after you issue the START TRACE command, you receive message DSNW133I, which indicates that the trace data is lost. This applies for destinations GTF, SRV, and SMF. You also receive this message for destinations OPn and OPX if START TRACE is not issued by an application program.

- GTF - The MVS generalized trace facility (GTF). The record identifier for records from DB2 is X′0FB9′.
- SMF - The system management facility. The SMF record type of DB2 trace records depends on the IFCID record, as shown in the following list:

IFCID Record	SMF Record Type
1 (SYSTEM SERVICES STATISTICS)	100
2 (DATABASE SERVICES STATISTICS)	100
3 (AGENT ACCOUNTING)	101
202 (DYNAMIC SYSTEM PARAMETERS)	100
230 (DATA SHARING GLOBAL STATISTICS)	100
239 (AGENT ACCOUNTING OVERFLOW)	101
ALL OTHERS	102

- SRV - An exit to a user-written routine. For instructions and an example of how to write such a routine, see the macro DSNWVSER in library prefix SDSNMACS.
- OPn - A specific destination. n can be an integer from 1 to 8.
- OPX - A generic destination which uses the first free OPn slot.

You will most likely trace to DESTination GTF or SMF.

You can trace specific entries, for example:

- PLAN(plan-name, ...) which introduces a list of specific plans for which trace information is to be captured. The default traces all plans.
- AUTHID(authid, ...) which traces specific authid(s).
- CLASS(class integer, ...) traces specific classes and is dependant on the type of trace data you are collecting.

Tracing all Performance Class records and writing to GTF would be started as follows:

```
-START TRACE(PERFM) DEST(GTF) CLASS(*)
```

DB2 dumps can be processed using IPCS and VERBX DB2DATA will format the DB2 dump data.

Appendix A. System logger

This appendix has files related to the chapter on the system logger.

A.1 LOGR IXCMIAPU sample member

```
//IFBLSJCL JOB                                                           00005000
//* START OF SPECIFICATIONS *******************************************  00000100
//*                                                                   *  00000200
//* Member Name:  IFBLSJCL                                            *  00000300
//*                                                                   *  00000400
//*  Descriptive Name:                                                *  00000500
//*   Sample JCL to provide an example of using the System Logger     *  00000600
//*   utility to define the Logrec log stream to a sysplex.           *  00000700
//*                                                                   *  00000800
//* Function:                                                         *  00000900
//*   This JCL sample provides an example of running the System       *  00001000
//*   Logger utility (IXCMIAPU) to define the Logrec log stream       *  00001100
//*   in the logger inventory.                                        *  00001200
//*                                                                   *  00001300
//*   Note that the MAXBUFSIZE parameter must have at least 4068 @L1C*  00005800
//*   specified, or Logrec will not be able to write to the Log       *  00001300
//*   stream.                                                         *  00001300
//*                                                                   *  00001300
//*   The Logrec log stream name must be specified as                 *  00005800
//*   SYSPLEX.LOGREC.ALLRECS.                                         *  00001300
//*                                                                   *  00001300
//* Operation:                                                        *  00001400
//*   Make suggested modifications as described below and then        *  00001500
//*   submit the job.                                                 *  00001510
//*                                                                   *  00001600
//* Suggested Modifications:                                          *  00001700
//*   Provide the specifications that are relevant for your           *  00001800
//*   installation on the SYSIN DATA TYPE(LOGR) definition.           *  00001900
//*                                                                   *  00002000
//*   For example, the following parameters define the log stream     *  00002100
//*   data set attributes:                                            *  00002200
//*                                                                   *  00002300
//*    LS_DATACLAS(data class)        - Name of data class            *  00002400
//*    LS_MGMTCLAS(management class)  - Name of management class      *  00002600
//*    LS_STORCLAS(storage class)     - Name of storage class         *  00002800
//*                                                                   *  00003000
//* References:                                                       *  00003100
//*   See System Logger guidance information in Assembler Services    *  00003200
//*   Guide.                                                          *  00003300
//*                                                                   *  00003400
//*   IXGINVNT is an executable macro that can be used as a program  *  00003500
//*   interface to define, update and delete entries in the           *  00003600
//*   MVS System Logger inventory.                                    *  00003700
//*                                                                   *  00003800
//* Recovery Operations:  None                                        *  00003900
//*                                                                   *  00004000
//* Distribution Library:  ASAMPLIB                                   *  00004100
//*                                                                   *  00004200
//* Change Activity:                                                  *  00004300
//*                                                                   *  00004400
```

```
//*FLAG LINEITEM  FMID    DATE    ID    COMMENT                       *  00004500
//* $L0=LOGRC     HBB5520 941020  PDDZ: Logrec log stream             *  00004600
//* $L1=LOGR4     JBB6604 961024  PDDZ: AVGBUFSIZE/MAXBUFSIZE    @L1A*  00004600
//*                                                                   *  00004700
//* End of Specifications ********************************************  00004800
//*                                                                 ***  00004900
//DEFINE   EXEC PGM=IXCMIAPU                                             00005100
//SYSPRINT DD   SYSOUT=A                                                 00005200
//SYSIN    DD   *                                                        00005300
 DATA TYPE (LOGR)                                                        00005400
    DEFINE STRUCTURE NAME(LOGRECSTRUCTURE)                               00005500
           LOGSNUM(1)                                                    00005600
           AVGBUFSIZE(4068)                                              00005700
           MAXBUFSIZE(4068)                                              00005800
    DEFINE LOGSTREAM NAME(SYSPLEX.LOGREC.ALLRECS)                        00005900
           STRUCTNAME(LOGRECSTRUCTURE)                                   00006000
/*                                                                       00006100
```

Appendix B. ISGGREX0 sample exit

When some DASD devices are shared with one or more MVS systems outside a GRS complex, it is mandatory, for data integrity reasons, to have *all* RESERVE/DEQ requests for the devices result in a hardware reserve/release. For example, a generic entry in a CONVERSION RNL for QNAME(SYSVTOC) converts all VTOC serialization requests to global ENQ requests and cannot be tolerated in this environment.

The attached ISGGREX0 exit is based on the ISGGREXS exit in SYS1.SAMPLIB. The sample exit detects volumes shared by systems outside the GRS complex and prevents RESERVE conversion for those volumes.

The shared “special” volumes are defined to the exit by adding a specific entry in the RNL conversion table with an unused QNAME and the shared volser as the RNAME, for example:

```
RNLDEF RNL(CON) TYPE(SPECIFIC)
QNAME(HWRESERV)
RNAME(VOLSER)
```

When a RESERVE/DEQ request is issued, the exit gets control and verifies whether the target volume for the RESERVE/DEQ request matches the special QNAME(HWRESERV) RNAME volser. If a match is found, return code 4 is passed back to the caller. Return code 4 indicates that GRS should *not* convert the RESERVE request (a SYSTEMS scope ENQ and the hardware reserve are to be issued). The target volser is found using the UCB address passed in the input parameters. If the RESERVE/DEQ target volser was not one of the special QNAME(HWRESERV) RNAME(VOLSER) volumes, normal CONVERSION RNL scan processing takes place.

Because the CONVERSION RNL table is used, any change to the table can be activated through operator commands.

To install the exit see *OS/390 MVS Installation Exits*, SC28-1753.

Note: Because this sample exit is using a qname - rname combination that is not used by the system, it can be installed and activated without having to cold start the sysplex. The suggested installation sequence is:

- Assemble and link-edit the sample ISGGREX0 exit into the MVS nucleus.
- Stop and re-IPL all the MVS systems in the sysplex, one by one. It is not required to stop the sysplex.
- Update the RNL conversion table with specific entries, qname(HWRESERV) rname(volser), for shared volumes.
- If the shared volumes have VSAM catalog, update the RNL exclusion table with specific entries, qname(SYSIGGV2) rname(catname), for the catalogs. For more information see the following note.
- Check that the shared volumes have no activity, and that they be logically offline.
- Dynamically activate the updated RNL table using command `SET GRSRNL=xx`.

Note: The volumes dynamically added or deleted to the QNAME(HWRESRV) with the RNL update should not be in use; they should be offline.

Note: If the shared volumes have VSAM catalogs, it is also recommended to place specific SYSIGGV2 entries for the catalog names in SYSTEMS exclusion RNL.

When using specific entries for SYSIGGV2 and the catalog name is less then 20 bytes, pad it to 20 bytes with blanks; if the name is more than 20, pad it to 44 bytes. If you have a catalog naming convention in place, you can use catalog-name prefixes and generic entries. In the following example, the third entry can be used to replace the first two entries; a naming convention should be in place.

```
RNLDEF RNL(EXCL) TYPE(SPECIFIC)
QNAME(SYSIGGV2)
RNAME(′CATALOG.SHRICF1.VIODFPK                     ′) <-- padded to 44 bytes

RNLDEF RNL(EXCL) TYPE(SPECIFIC)
QNAME(SYSIGGV2)
RNAME(′CATALOG.SHRICF1.VIODFPK                     ′) <-- padded to 44 bytes

RNLDEF RNL(EXCL) TYPE(GENERIC)
QNAME(SYSIGGV2)
RNAME(CATALOG.SHRICF1)
```

Figure 78 on page 125 shows the conversion exit logic.

```
         TITLE ′ISGGREXO - GRS RESOURCE EXIT ROUTINE′
ISGGREXO CSECT
*/*  START OF SPECIFICATIONS ****
*-----------------------------------------------------------------*
* ALWAYS RESERVE FOR VOLUMES SHARED WITH MVS SYSTEMS OUTSIDE      *
* THE GRS RING.                                                   *
*                                                                 *
* THIS ISGGREXO EXAMPLE IS BASED ON THE ISGGREXS IN THE           *
* SYS1.SAMPLIB, WITH SUPPORT FOR SUPPRESSION OF TEMPORARY         *
* DATA SETS IN GLOBAL SHARING REMOVED.                            *
*                                                                 *
* THE LINE OF CODE MODIFIED OR ADDED ARE INDICATED WITH      *AAA *
*                                                                 *
*-----------------------------------------------------------------*
* LOGIC:                                                          *
*                                                                 *
* IT IS MANDATORY, TO SUPPORT SHARED DASD WITH MVS SYSTEMS        *
* OUTSIDE GRS, THAT ALL RESERVE REQUESTS THAT ADDRESS THESE       *
* VOLUMES RESULT IN AN H/W RESERVE WHATEVER THE RESOURCE NAME IS. *
*                                                                 *
* THE SAME RESOURCES, THAT ADDRESS OTHER VOLUMES, SHOULD HAVE THE *
* POSSIBILITY TO BE FILTERED THROUGH THE CONVERSION RNLS AND HAVE *
* THE HARDWARE RESERVE ELIMINATED.                                *
*                                                                 *
* BY ADDING TO THE CONVERSION TABLE A QNAME NOT USED BY THE       *
* SYSTEM AND RNAMES THAT IDENTIFY THE VOLUMES TO SHARE OUTSIDE    *
* GRS, THE RESERVE REQUEST FOR THE VOLUMES INCLUDED IN THE        *
* FOLLOWING DEFINITION WILL ALWAYS RESULT IN A HARDWARE RESERVE.  *
*                                                                 *
*      RNLDEF RNL(CON) TYPE(SPECIFIC)                             *
*      QNAME(HWRESERV)                                            *
*      RNAME(VOLSER)                                              *
*                                                                 *
* ISGGRCEXO RECEIVES CONTROL FOR ALL RESERVE/DEQ  REQUESTS        *
* (SYSTEMS+H/W RESERVE), LOCATES THE UCBVOLI (VOLSER) AND CHECKS  *
* IF THE RESOURCE ′HWRESERV′ ′VOLSER′ IS PRESENT IN THE ′CON′     *
* TABLE. IF FOUND RETURNS RC=4 TO GRS THAT LEAVES THE RESOURCE    *
* AS IS (SYSTEMS+H/W RESERVE). IF NOT NORMAL RNL SCAN IS RESUMED. *
*                                                                 *
* BECAUSE THE CONVERSION RNL TABLE IS USED, ANY CHANGE TO THE     *
* TABLE CAN BE ACTIVATED THROUGH OPERATOR COMMANDS.               *
*                                                                 *
* NOTE: THE VOLUME(S) BEHIND THE QNAME(HWRESERV) SHOULD NOT BE    *
* DYNAMICALLY CHANGED UNLESS THE DEVICES ARE OFFLINE.             *
*                                                                 *
* THE NAME ′HWRESERV′ IS HARD-CODED AND IS DEFINED AT LABEL       *
* HRDWNAME.                                                       *
*                                                                 *
*-----------------------------------------------------------------*
*                 CONVERSION LIST EXAMPLE                         *
*-----------------------------------------------------------------*
*                                                                 *
* RNLDEF RNL(CON) TYPE(GENERIC)                                   *
* QNAME(SYSVTOC)                                                  *
*                                                                 *
* RNLDEF RNL(CON) TYPE(GENERIC)                                   *
* QNAME(SYSIGGV2)                                                 *
*                                                                 *
* RNLDEF RNL(CON) TYPE(SPECIFIC)                                  *
* QNAME(HWRESERV)                     /*SPECIAL NAME*/            *
* RNAME(XA9RES)                       /*ALWAYS RESERV  XA9RES*/   *
*                                                                 *
* RNLDEF RNL(CON) TYPE(SPECIFIC)                                  *
* QNAME(HWRESERV)                     /*SPECIAL NAME*/            *
* RNAME(CIX321)                       /*ALWAYS RESERV  CIX321*/   *
*                                                                 *
*******************************************************************
***                        END OF                                 *
***   RESERVE CONVERSION RESOURCE NAME LIST - SAMPLE              *
*******************************************************************
*---------------------------------------------------------------------
*
*
*01*  MODULE-NAME = ISGGREX0
*
*02*    CSECT-NAME = ISGGREX0
*
*01*  DESCRIPTIVE-NAME = GRS RESOURCE EXIT ROUTINE
*
*01*  COPYRIGHT =
*        5740-XC6 COPYRIGHT IBM CORP 1981,
*        LICENSED MATERIAL-PROGRAM, PROPERTY OF IBM,
*        REFER TO COPYRIGHT INSTRUCTIONS FORM NUMBER G120-2083.
*
*01* STATUS = OS/VS2 HBB2102
*
*01*  FUNCTION =
*                TO SCAN THE INPUT RESOURCE NAME LIST (RNL) FOR
*                THE RESOURCE NAME SPECIFIED IN THE INPUT PEL.
*                THE RETURN CODE INDICATES WHETHER OR NOT THE
*                INPUT RESOURCE NAME IS CONTAINED IN THE RNL.
*
*02*    OPERATION =
*                   THE INPUT RESOURCE NAME LIST IS SEARCHED AS
*                   FOLLOWS:
*
*                   1.  COMPARE THE INPUT ARGUMENT TO THE RNL ENTRY.
*
*                   2.  IF A MATCH IS FOUND, SET INDICATIVE RETURN
*                       CODE.
*
*                   3.  IF A MATCH IS NOT FOUND, INDEX TO NEXT RNL
*                       ENTRY AND CONTINUE SCAN.
*
*                   4.  IF THE ENTIRE RNL IS SEARCHED WITHOUT A
*                       MATCH, SET INDICATIVE RETURN CODE.
*
*01*  NOTES =
*             1.  TO FACILITATE REPLACEMENT OF THE RESOURCE EXITS
*                 PROVIDED BY GRS, THIS ROUTINE HAS BEEN DESIGNED
*                 TO SCAN ANY OF THE 3 GRS RNL'S. THOUGH ONE
*                 ROUTINE IS PROVIDED TO PERFORM THE SCAN, IT IS
*                 INVOKED USING 3 DIFFERENT EXTERNAL NAMES:
*
*                 ISGGSIEX - INVOKED TO SCAN THE SYSTEM (SCOPE)
*                            INCLUSION RESOURCE NAME LIST.
*
*                 ISGGRCEX - INVOKED TO SCAN THE RESERVE
*                            CONVERSION RESOURCE NAME LIST.
*
*                 ISGGSEEX - INVOKED TO SCAN THE SYSTEMS (SCOPE)
*                            EXCLUSION RESOURCE NAME LIST.
*
*                 EACH OF THESE EXTERNAL ENTRY POINT NAMES AND
*                 ITS CORRESPONDING RNL ADDRESS HAS BEEN
*                 DEFINED IN THE GVT.  THE ENQ/DEQ/RESERVE MAINLINE
*                 ROUTINE INVOKES THE APPROPRIATE EXIT ROUTINE
*                 (E.G., RESERVE PROCESSING INVOKES ISGGRCEX)
*                 ALTHOUGH EACH ENTRY POINT MAPS TO THE COMMON
*                 SCAN ROUTINE PROVIDED BY GRS.  THEREFORE THE
*                 INSTALLATION CAN REPLACE THE EXIT ROUTINE
*                 AND/OR RNL(S) WITHOUT CHANGES TO THE GVT OR
*                 THE ENQ/DEQ/RESERVE MAINLINE ROUTINE.
*             2.  MODULE ISGGREX0 CAN BE REPLACED WITH WHATEVER
*                 AMODE OR RMODE THE INSTALLATION WISHES. THE
*                 DEFAULTS USED ARE AMODE(31) AND RMODE(ANY).  @G860PWE
*
*             3.  EACH PROCESSOR IN THE GRS RING MUST HAVE
*                 IDENTICAL RNL'S.  ALSO, THE EXIT ROUTINE
*                 IN EACH GRS SYSTEM MUST YIELD THE SAME SCAN
*                 RESULTS.  REFERENCE ISGCQMRG FOR FURTHER
*                 INFORMATION.
*
*             4.  SERIALIZATION WHEN INVOKED AS ISGGSIEX,
*                 ISGGSEEX, OR ISGGRCEX IS THE LOCAL LOCK
*                 OF THE INVOKER'S ADDRESS SPACE AND
*                 CMSEQDQ.
```

```
*
*02*    DEPENDENCIES =
*                1.  THE RECOVERY ROUTINE FOR THIS MODULE,
*                    ISGGFRRO, DEPENDS ON THE FOLLOWING LABELS
*                    TO APPEAR IN THIS MODULE:
*                       ISGGREXO  CSECT NAME
*                       ISGGREXE  END OF MODULE ISGGREXO
*                       ISGGREXM  LABEL ON THE MODID MACRO
*
*                    IF THE USER REPLACES THIS MODULE, IT MUST
*                    BE INSURED THAT THE ABOVE LABELS APPEAR
*                    IN THE APPROPRIATE LOCATIONS IN THE
*                    EXIT ROUTINE.
*
*                2.  THIS MODULE WILL FUNCTION ON ALL HARDWARE
*                    AND SOFTWARE CONFIGURATIONS WHICH SATISFY
*                    THE REQUIREMENTS OF THE PRODUCT IDENTIFIED
*                    IN THE STATUS FIELD.
*                3.  ANY INPUT ADDRESSES TO ISGGREXO MUST BE
*                    24-BIT ADDRESSES TO ALLOW THE MODULE TO
*                    BE REPLACED IN ANY AMODE OR ANY RMODE AN
*                    INSTALLATION WISHES.                      @G860PWE
*
*02*    CHARACTER-CODE-DEPENDENCIES = EBCDIC CHARACTER SET
*
*02*    RESTRICTIONS = NONE.
*
*02*    REGISTER-CONVENTIONS =
*
*03*      REGISTERS-SAVED =
*                           R0 - R12, R14, R15
*
*03*      REGISTER-USAGE =
*                 R0     = CONTAINS RNL ADDRESS ON ENTRY. CONTAINS
*                          RETURN CODE WHILE SCANNING LISTS
*                 R1     = ADDRESS OF A PEL ON ENTRY
*                 R2     = USED TO ADDRESS RNLES
*                 R3-R12 = USAGE UNPREDICTABLE
*                 R13    = SAVE AREA ADDRESS
*                 R14    = RETURN ADDRESS
*                 R15    = ENTRY POINT ADDRESS/BASE REGISTER.
*                          UPON EXIT, RETURN CODE.
*
*03*      REGISTERS-RESTORED =
*                              R0 - R12, R14
*
*02*    PATCH-LABEL = NONE (NUCLEUS RESIDENT - USE IEAPATCH)
*
*01*  MODULE-TYPE = PROCEDURE
*
*02*    PROCESSOR = PL/S-III
*
*02*    MODULE-SIZE = SEE EXTERNAL SYMBOL DICTIONARY
*
*02*    ATTRIBUTES =
*
*03*      LOCATION      = NUCLEUS
*03*      STATE         = SUPERVISOR
*03*      KEY           = 0
*03*      MODE          =
*
*04*        WORK UNIT = TASK
*04*        HASID     = ANY
*04*        PASID     = ANY
*04*        SASID     = ANY
*04*        SAC       = ON OR OFF
*04*        AMODE     = 31
*04*        RMODE     = ANY
*
*03*      SERIALIZATION = SERIALIZATION WHEN INVODED AS ISGGSIEX,
*                         ISGGSEEX, OR ISGGRCEX IS THE LOCAL LOCK
*                         OF THE INVOKER'S ADDRESS SPACE AND
*                         CMSEQDQ.
*
*03*      TYPE          = REENTRANT
*
*01*  ENTRY-POINT = ISGGSIEX (ENTRY POINT DEFINED IN GVTGSIEX)
*
*02*    PURPOSE = TO DETERMINE WHETHER THE INPUT RESOURCE NAME
*                 SHOULD BE INCLUDED IN GLOBAL SHARING BY
*                 SCANNING THE SYSTEM (SCOPE) INCLUSION RNL.
*
*03*      OPERATION = SEE MODULE OPERATION SECTION
```

```
*
*02*    LINKAGE = BALR
*
*03*      CALLERS = ISGGQWBI, ISGLNQDQ
*
*03*      ENTRY-REGISTERS =
*                 R0     = ADDRESS OF SYSTEM INCLUSION
*                          RESOURCE NAME LIST
*                 R1     = ADDRESS OF A PEL
*                 R2-R12 = UNDEFINED
*                 R13    = SAVEAREA ADDRESS
*                 R14    = RETURN ADDRESS
*                 R15    = ENTRY POINT ADDRESS
*
*02*    INPUT =
*              R0 = ADDRESS OF RNL TO BE SEARCHED
*              R1 = PEL ENTRY  CONTAINING RESOURCE NAME TO BE
*                   USED AS THE SEARCH ARGUMENT.
*
*02*    OUTPUT =
*               R15 = 0 - NAME FOUND
*                     4 - NAME NOT FOUND
*
*02*    EXIT-NORMAL = RETURN TO CALLER VIA BR14.
*
*03*      CONDITIONS = INPUT LIST SCANNED AS REQUESTED.
*
*04*        EXIT-REGISTERS =
*                  R0-R12 = AS ON INPUT
*                  R13    = SAVEAREA ADDRESS
*                  R14    = RETURN ADDRESS
*                  R15    = RETURN CODE
*
*03*      RETURN-CODES =
*                       R15 = 0 - NAME FOUND
*                             4 - NAME NOT FOUND
*
*02*    EXIT-ERROR = NONE
*
*02*    WAIT-STATE-CODES = NONE
*
*01*  ENTRY-POINT = ISGGRCEX (ENTRY POINT DEFINED IN GVTGRCEX)
*
*02*    PURPOSE = TO DETERMINE WHETHER THE INPUT RESOURCE NAME
*                SHOULD BE CONVERTED FROM A HARDWARE RESERVE
*                RESOURCE NAME TO A GRS CONTROLLED GLOBAL ENQ
*                RESOURCE NAME BY SCANNING THE RESERVE CONVERSION
*                RNL.
*
*03*      OPERATION = SEE MODULE OPERATION SECTION.
*
*02*    LINKAGE = BALR
*
*03*      CALLERS = ISGGQWBI
*
*03*      ENTRY-REGISTERS =
*                 R0     = ADDRESS OF SYSTEM RESERVE CONVERSION
*                          RESOURCE NAME LIST
*                 R1     = ADDRESS OF A PEL
*                 R2-R12 = UNDEFINED
*                 R13    = SAVEAREA ADDRESS
*                 R14    = RETURN ADDRESS
*                 R15    = ENTRY POINT ADDRESS
*
*02*    INPUT =
*              R0 = ADDRESS OF RNL TO BE SEARCHED
*              R1 = PEL ENTRY  CONTAINING RESOURCE NAME TO BE
*                   USED AS THE SEARCH ARGUMENT.
*
*02*    OUTPUT =
*               R15 = 0 - NAME FOUND
*                     4 - NAME NOT FOUND
*
*02*    EXIT-NORMAL = RETURN TO CALLER VIA BR14.
*
*03*      CONDITIONS = INPUT LIST SCANNED AS REQUESTED.
*
*04*        EXIT-REGISTERS =
*                  R0-R12 = AS ON INPUT
*                  R13    = SAVEAREA ADDRESS
*                  R14    = RETURN ADDRESS
*                  R15    = RETURN CODE.
*
```

```
*03*      RETURN-CODES =
*                        R15 = 0 - NAME FOUND
*                              4 - NAME NOT FOUND
*
*02*    EXIT-ERROR = NONE
*
*02*    WAIT-STATE-CODES = NONE
*
*01*  ENTRY-POINT = ISGGSEEX  (ENTRY POINT DEFINED IN GVTGSEEX)
*
*02*    PURPOSE = TO DETERMINE WHETHER THE INPUT RESOURCE NAME
*                 SHOULD BE EXCLUDED FROM GLOBAL SHARING BY
*                 SCANNING THE SYSTEMS (SCOPE) EXCLUSION RNL.
*
*03*      OPERATION = SEE MODULE OPERATION SECTION.
*
*02*    LINKAGE = BALR
*
*03*      CALLERS = ISGGQWBI
*
*03*      ENTRY-REGISTERS =
*                  R0     = ADDRESS OF SYSTEMS EXCLUSION RESOURCE
*                           NAME LIST
*                  R1     = ADDRESS OF A PEL
*                  R2-R12 = UNDEFINED
*                  R13    = SAVEAREA ADDRESS
*                  R14    = RETURN ADDRESS
*                  R15    = ENTRY POINT ADDRESS
*
*02*    INPUT =
*               R0 = ADDRESS OF RNL TO BE SEARCHED
*               R1 = PEL ENTRY  CONTAINING RESOURCE NAME TO BE
*                    USED AS THE SEARCH ARGUMENT.
*
*02*    OUTPUT =
*                R15 = 0 - NAME FOUND
*                      4 - NAME NOT FOUND
*
*02*    EXIT-NORMAL = RETURN TO CALLER VIA BR14.
*
*03*      CONDITIONS = INPUT LIST SCANNED AS REQUESTED.
*
*04*        EXIT-REGISTERS =
*                   R0-R12 = AS ON INPUT
*                   R13    = SAVEAREA ADDRESS
*                   R14    = RETURN ADDRESS
*                   R15    = RETURN CODE.
*
*03*      RETURN-CODES =
*                        R15 = 0 - NAME FOUND
*                              4 - NAME NOT FOUND
*
*02*    EXIT-ERROR = NONE
*
*02*    WAIT-STATE-CODES = NONE
*
*01*  EXTERNAL-REFERENCES = NONE.
*
*02*    ROUTINES = NONE.
*
*02*    DATA-AREAS = NONE
*
*02*    CONTROL-BLOCKS =
*                        PEL       R
*                        RNLE      R
*
*01*  TABLES =
*              1. SYSTEMS (SCOPE) INCLUSION RNL (ISGGIRNL)
*              2. SYSTEM  (SCOPE) EXCLUSION RNL (ISGGERNL)
*              3. RESERVE CONVERSION RNL        (ISGGCRNL)
*
*01*  MACROS-EXECUTABLE =
*                           MODID
*
*02*    SERIALIZATION = NONE.
*
*01*  CHANGE-ACTIVITY = SUPPORTS THE FOLLOWING PRODUCTS:
*                        $L0=FUNCT,JBB1326,810323,PDYC: VERSION OF @L0A
*                                 ISGGREX0 FOR SAMPLIB             @L0A
*                        HBB2102  NEW VERSION OF ISGGREX0 PLUS @G860PYC
*                                 SUPPORT FOR TEMPORARY D.S.   @G860PYC
*
*                       SUPPORTS THE FOLLOWING PTMS:
```

```
*                        PBB0618 ADD RECOVERY SUPPORT          @ZMB0618
*                        PBB0854 REMOVE ISGGRNLV ENTRY PT.     @ZMB0854
*                        PCC2507 UPDATE COPYRIGHT FIELD AND    @ZMC2507
*                                ADD A STATUS FIELD            @ZMC2507
*
*01*  MESSAGES = NONE.
*
*01*  ABEND-CODES = NONE.
*
*01*  SYSGEN = INCLUDED FROM AOSC5 INTO IEANUC01 BY SGISG300
*
**** END OF SPECIFICATIONS **                                        */
         EJECT
***********************************************************************
*                                                                     *
*        REGISTER ASSIGNMENTS                                         *
*                                                                     *
***********************************************************************
         SPACE
RNLSTART EQU   0                       UPON ENTRY, START OF RNL
RTRNCODE EQU   0                       MATCH/NO-MATCH INDICATOR
PELPTR   EQU   1                       ADDRESS OF THE PEL
RNLEPTR  EQU   2                       POINTER TO AN RNL ENTRY (RNLE)
CHARPTR  EQU   3                       ADDR OF CHARACTER BEING TESTED
FLENRNLE EQU   3                       LENGTH OF FIXED PART OF AN RNLE
RNAMELEN EQU   3                       LENGTH OF RNAME
WORKREG  EQU   12                      LENGTH OF FIXED + VARIABLE
REG13    EQU   13                      SAVE AREA ADDRESS
REG14    EQU   14                      RETURN ADDRESS
REG15    EQU   15                      USED AS BASE REG UNTIL EXIT,
*                                      THEN CONTAINS THE RETURN CODE
R0       EQU   0                       START DI RNL
R1       EQU   1                       ADDRESS OF PEL
R2       EQU   2                       POINTER TO AN RNLE
R3       EQU   3                       LENGTH OF RNAME
R4       EQU   4
R5       EQU   5
R6       EQU   6
R7       EQU   7
R8       EQU   8
R9       EQU   9
R10      EQU   10
R11      EQU   11
R12      EQU   12
R13      EQU   13                      SAVE AREA POINTER
R14      EQU   14                      RETURN ADDRESS
R15      EQU   15                      BASE REGISTER UNTIL EXIT
         SPACE 5
***********************************************************************
*                                                                     *
*        CONSTANTS                                                    *
*                                                                     *
***********************************************************************
         SPACE
FOUND    EQU   0                       RESOURCE NAME IS IN THE RNL
NOTFOUND EQU   4                       RESOURCE NAME IS NOT IN THE RNL
ZERO     EQU   0                       USED FOR LENGTH TEST
         EJECT
*
ISGGREX0 AMODE 31                                              @G860PYC
ISGGREX0 RMODE ANY                                             @G860PYC
         MODID BR=NO,MODLBL=ISGGREXM
MAINENT  DS    0H
         USING *,15
         ENTRY ISGGSIEX
         ENTRY ISGGSEEX
         ENTRY ISGGRCEX
         ENTRY ISGGREXE                END OF MODULE
         ENTRY ISGGREXM                MODULE INFORMATION
*-------------------------------------------------------------  *AAA
* CODE TO SUPPORT ’ALWAYS RESERVE’ FOR VOLUMES SHARED WITH MVS  *AAA
* SYSTEMS OUTSIDE THE GRS COMPLEX.                              *AAA
*-------------------------------------------------------------  *AAA
* CONVERSION ENTRY                                              *AAA
*                                                               *AAA
* FUNCTION                                                      *AAA
*  CHECK REQUEST FOR RESERVE ===>SYSTEMS + H/W-RESERVE          *AAA
*  FIND VOLSER FROM UCB POINTED BY PEL                          *AAA
*  CHECK IF CONVERSION TABLE HAS A SPECIFIC ENTRY FOR           *AAA
*                                                               *AAA
*        QNAME(HWRESERV)                                        *AAA
*        RNAME(VOLSER)                                          *AAA
*                                                               *AAA
*  IF MATCH FOUND RETURN CODE = 4 ==> SYSTEMS + H/W-RESERVE     *AAA
*  IF NOT, RESTORE REGISTERS AND CONTINUE NORMALLY              *AAA
*-------------------------------------------------------------  *AAA
ISGGRCEX DS    OH                *AAA  SCAN RESERVE CONVERSION RNL
         SPACE
         STM   14,12,12(13)      *AAA  SAVE ENTRY REGS
         USING RNLE,RNLEPTR      *AAA  ADDRESSABILITY FOR RNL ENTRIES
         USING PEL,PELPTR        *AAA  PEL ENTRY ADDRESSABILITY
         LR    RNLEPTR,RNLSTART  *AAA  SET RNLE PTR TO START OF RNL
         LA    RTRNCODE,NOTFOUND *AAA  RO = 4   NO MATCH
*
         TM    PELFLAG,PELSCPE2  *AAA  SYSTEMS+UCB
         BZ    NOHARDW           *AAA
         L     R1,PELXUCBA       *AAA  UCB POINTER CLEAN,
*                                      HIGH BYTE NOT USED
         DROP  PELPTR            *AAA
*
* R1 = UCB POINTER USED FOR  RNAME (VOLSER)
*
         SPACE
COMPRNL1 EQU   *                 *AAA  COMPARE PEL NAME TO RNL ENTRY
         TM    RNLEFLGS,RNLELAST *AAA  AT END OF THE RNL ?
         BO    NOHARDW           *AAA  YES => RNL SCAN DONE
         SPACE
* COMPARE QNAME HWRESERV   TO THE ONE IN THE RNLE
         CLC   HRDWNAME,RNLEQNME *AAA  CHECK IF QNAME HWRESERV
         BNE   NEXTRNL1          *AAA  NO => GET NEXT RNL ENTRY
         SPACE
* COMPARE THE VOLSER
         SLR   RNAMELEN,RNAMELEN *AAA  CLEAR WORK REG
         IC    RNAMELEN,RNLERNML *AAA  GET LENGTH OF RNAME IN THE RNLE
         BCTR  RNAMELEN,ZERO     *AAA  ADJUST LENGTH
         EX    RNAMELEN,COMPRNM1 *AAA  COMPARE THE RNAME (VOLSER)
         BNE   NEXTRNL1          *AAA  UNEQUAL => GET NEXT RNLE
         SPACE
* INDICATE RESOURCE NAME FOUND IN THE RNL
         LA    RTRNCODE,NOTFOUND *AAA  INDICATE NO CONVERSION
*                                      FOR THIS UCB
         B     MODEXIT           *AAA  EXIT PROCESSING COMPLETE
         SPACE
* GET THE ADDRESS OF THE NEXT RNL ENTRY
NEXTRNL1 EQU   *
         LA    FLENRNLE,RNLERNME-RNLE  LENGTH OF FIXED PART OF RNLE
         SLR   WORKREG,WORKREG   *AAA  CLEAR WORK REG
         IC    WORKREG,RNLERNML  *AAA  GET RNAME LENGTH (VARIABLE)
         ALR   WORKREG,FLENRNLE  *AAA  ADD FIXED + VARIABLE LENGTHS
         ALR   RNLEPTR,WORKREG   *AAA  GET ADDRESS OF NEXT RNL ENTRY
         B     COMPRNL1          *AAA  CHECK THE NEW RNL ENTRY
         EJECT
*---------------------------------------------------------------------
* RO, R1, DA RIPRISTINARE
* R13 SAVE AREA, R14, RETURN ADDRESS, R15 BASE
*---------------------------------------------------------------------
NOHARDW  EQU   *                 *AAA
         DROP  RNLEPTR
         LM    RO,R12,20(R13)    *AAA  RIPRISTINO REGISTRI
         B     SECENTRY          *AAA  A CODIFICA NORMALE
*
*--FINE MODIFICA SUPPORTO DISCHI CONDIVISI FUORI SYSPLEX  *AAA-------
***********************************************************************
         EJECT
ISGGSIEX DS    OH                      SCAN INCLUSION RNL
ISGGSEEX EQU   *                       SCAN EXCLUSION RNL
*SGGRCEX EQU   *                       SCAN RESERVE CONVERSION RNL *AAA
         SPACE 3
***********************************************************************
*                                                                     *
*        LOGIC FLOW FOR ISGGREXO                                      *
*                                                                     *
***********************************************************************
*/*                                                                  */
*/*++ ’ISGGREXO’: ENTRY TO RNL EXIT PROCESSING                       */
*/*++ ESTABLISH ADDRESSABILITY                                       */
*/*++ SET RETURN CODE TO NOT-FOUND                                   */
*/*++ DO WHILE MATCH NOT FOUND AND NOT LAST ENTRY IN THE RNL         */
*/*++   IF THE PEL EXTENSION QNAME EQUALS THE QNAME IN THE RNL ENTRY */
*/*++     IF THE RNL ENTRY IS GENERIC                                */
*/*++       IF THE LENGTH OF THE RNAME IN THE RNL ENTRY IS ZERO      */
*/*++         SET RETURN CODE TO FOUND                               */
*/*++       ELSE  (RNLE RNAME LENGTH IS NOT ZERO)                    */
*/*++         IF THE PEL RNAME LENGTH IS GREATER THAN OR EQUAL TO
*                THE RNAME LENGTH IN THE RNL ENTRY                   */
*/*++           IF THE RNAME IN THE RNLE MATCHES THE ONE IN THE PEL  */
```

```
*/*++             SET RETURN CODE TO FOUND                            */
*/*++           ENDIF    (END OF RNAME COMPARISON)                    */
*/*++         ENDIF    (END OF COMPARISON OF RNAME LENGTHS)           */
*/*++       ENDIF  (END OF CHECK FOR RNAME LENGTH OF ZERO)            */
*/*++     ELSE  (RNL ENTRY IS NON-GENERIC)                            */
*/*++       IF THE RNAME LENGTH IN THE RNL EQUALS THE RNAME LENGTH
*              IN THE PEL                                             */
*/*++         IF THE RNAME IN THE RNLE MATCHES THE ONE IN THE PEL     */
*/*++           SET RETURN CODE TO FOUND                              */
*/*++         ENDIF  (END OF RNAME COMPARISON)                        */
*/*++       ENDIF  (END OF COMPARISON OF RNAME LENGTHS)               */
*/*++     ENDIF  (END OF GENERIC/NON-GENERIC CHECK)
*/*++   ENDIF  (END OF QNAME COMPARISON)                              */
*/*++   GET NEXT RNL ENTRY                                            */
*/*++ ENDDO  (REPEAT SEQUENCE UNTIL MATCH FOUND OR END OF RNL)        */
*/*++ RETURN                                                          */
*/*++ END ’ISGGREX0’                                                  */
*/*                                                                   */
*/*********************************************************************/
         EJECT
************************************************************************
*                                                                      *
*  1. SAVE ENTRY REGISTERS                                             *
*                                                                      *
*  2. ESTABLISH CONTROL BLOCK ADDRESSABILITY                           *
*                                                                      *
*  3. CHECK FOR END OF RNL.  IF END, GO TO STEP 15.                    *
*                                                                      *
*  4. COMPARE QNAME IN PEL TO RNLE.  IF NOT EQUAL, GO TO STEP 14.      *
*                                                                      *
*  5. CHECK FOR GENERIC ENTRY IN THE RNLE.  IF NOT, GO TO STEP 11.     *
*                                                                      *
*  6. CHECK FOR AN RNAME PRESENT IN THE RNLE.  IF YES, GO TO STEP 8.   *
*                                                                      *
*  7. QNAMES MATCH + GENERIC ENTRY + NO RNAME IN THE RNLE              *
*     => THE NAME IN THE PEL MATCHES.  GO TO STEP 24.                  *
*                                                                      *
************************************************************************
         SPACE
SECENTRY EQU   *                                             *AAA
         BALR  R15,0                                         *AAA
         DROP  R15
         USING *,R15                                         *AAA
         STM   14,12,12(13)            SAVE ENTRY REGS
         USING RNLE,RNLEPTR            ADDRESSABILITY FOR RNL ENTRIES
         USING PEL,PELPTR              PEL ENTRY ADDRESSABILITY
         LR    RNLEPTR,RNLSTART        SET RNLE PTR TO START OF RNL
         LA    RTRNCODE,NOTFOUND       ASSUME NO MATCH WILL BE FOUND
         SPACE
COMPRNLE EQU   *                       COMPARE PEL NAME TO RNL ENTRY
         TM    RNLEFLGS,RNLELAST       AT END OF THE RNL ?
         BO    ENDOFRNL                YES => RNL SCAN DONE
         SPACE
* COMPARE QNAME IN THE PEL TO THE ONE IN THE RNLE
         CLC   PELXQNME,RNLEQNME       CHECK IF QNAMES MATCH
         BNE   NEXTRNLE                NO => GET NEXT RNL ENTRY
         SPACE
* CHECK FOR GENERIC ENTRY IN THE RNL
         TM    RNLEFLGS,RNLEGENR       GENERIC ENTRY IN THE RNL ?
         BNO   NONGENER                NO => HANDLE NON-GENERIC ENTRY
         SPACE
* CHECK FOR AN RNAME PRESENT IN THE RNLE
         CLI   RNLERNML,ZERO           IS THE RNLE RNAME LENGTH ZERO ?
         BNE   CHKLEN                  NO => CHECK RNAME LENGTHS
         SPACE
* INDICATE RESOURCE NAME IS IN THE RNL
         LA    RTRNCODE,FOUND          MATCH HAS BEEN FOUND
         B     MODEXIT                 EXIT PROCESSING COMPLETE
         EJECT
************************************************************************
*                                                                      *
*  8. IS THE LENGTH OF THE RNAME IN THE PEL TOO SHORT FOR THE GENERIC *
*     ENTRY IN THE RNLE?  IF SO, GO TO STEP 14.                        *
*                                                                      *
*  9. CHECK IF THE RNAMES MATCH.  IF NOT, GO TO STEP 14.               *
*                                                                      *
* 10. RNAMES MATCH.  NAME FOUND IN THE RNL.  GO TO STEP 24.            *
*                                                                      *
************************************************************************
         SPACE
CHKLEN   EQU   *
* CHECK IF THE LENGTH OF THE RNAME IN THE PEL IS TOO SHORT
         CLC   PELMILEN,RNLERNML       PEL RNAME LEN >= RNLE RNAME LEN?
         BL    NEXTRNLE                NO => NO MATCH, GET NEXT RNLE
         SPACE
* CHECK IF THE RNAMES MATCH
         SLR   RNAMELEN,RNAMELEN       CLEAR WORK REG
         IC    RNAMELEN,RNLERNML       GET LENGTH OF RNAME IN THE RNLE
         BCTR  RNAMELEN,ZERO           ADJUST LENGTH
         EX    RNAMELEN,COMPRNME       COMPARE THE RNAMES
         BNE   NEXTRNLE                UNEQUAL => GET NEXT RNLE
         SPACE
* INDICATE RESOURCE NAME FOUND IN THE RNL
         LA    RTRNCODE,FOUND          MATCH HAS BEEN FOUND
         B     MODEXIT                 EXIT PROCESSING COMPLETE
         EJECT
***********************************************************************
*                                                                     *
*        NON-GENERIC ENTRY PROCESSING                                 *
*                                                                     *
* 11. CHECK IF RNAME LENGTHS ARE EQUAL.  IF NOT, GO TO STEP 14.       *
*                                                                     *
* 12. COMPARE RNAMES.  IF UNEQUAL, GO TO STEP 14.                     *
*                                                                     *
* 13. RNAMES MATCH.  NAME FOUND IN THE RNL.  GO TO STEP 24.           *
*                                                                     *
*        GET NEXT RNL ENTRY                                           *
*                                                                     *
* 14. SKIP OVER THE RNL ENTRY JUST TESTED.  GO TO STEP 3.             *
*                                                                     *
***********************************************************************
         SPACE
NONGENER EQU   *                       HANDLE NON-GENERIC RNL ENTRY
* CHECK IF RNAME LENGTHS ARE EQUAL
         CLC   PELMILEN,RNLERNML       PEL RNAME LEN = RNLE RNAME LEN ?
         BNE   NEXTRNLE                NO => NO MATCH, GET NEXT RNLE
         SPACE
* COMPARE THE RNAMES
         SLR   RNAMELEN,RNAMELEN       CLEAR WORK REG
         IC    RNAMELEN,RNLERNML       GET LENGTH OF RNAME IN THE RNLE
         BCTR  RNAMELEN,ZERO           ADJUST LENGTH
         EX    RNAMELEN,COMPRNME       COMPARE THE RNAMES
         BNE   NEXTRNLE                UNEQUAL => GET NEXT RNLE
         SPACE
* INDICATE RESOURCE NAME FOUND IN THE RNL
         LA    RTRNCODE,FOUND          MATCH HAS BEEN FOUND
         B     MODEXIT                 EXIT PROCESSING COMPLETE
         SPACE
* GET THE ADDRESS OF THE NEXT RNL ENTRY
NEXTRNLE EQU   *
         LA    FLENRNLE,RNLERNME-RNLE  LENGTH OF FIXED PART OF RNLE
         SLR   WORKREG,WORKREG         CLEAR WORK REG
         IC    WORKREG,RNLERNML        GET RNAME LENGTH (VARIABLE)
         ALR   WORKREG,FLENRNLE        ADD FIXED + VARIABLE LENGTHS
         ALR   RNLEPTR,WORKREG         GET ADDRESS OF NEXT RNL ENTRY
         B     COMPRNLE                CHECK THE NEW RNL ENTRY
         EJECT
***********************************************************************
*                                                                     *
*  CONTROL ARRIVES AT ENDOFRNL ONLY WHEN THE ENTIRE RNL HAS BEEN      *
*  SCANNED AND NO ENTRY WAS FOUND THAT MATCHED THE RESOURCE NAME      *
*  IN THE PEL.                                                        *
*                                                                     *
         SPACE
ENDOFRNL EQU   *
***********************************************************************
*                                                                     *
* 24. RETURN TO CALLER.        M O D E X I T                          *
*                                                                     *
***********************************************************************
         SPACE
MODEXIT  EQU   *
         LR    REG15,RTRNCODE          GET RETURN CODE INTO REG 15
         L     14,12(,REG13)           RECOVER THE RETURN ADDRESS
         LM    0,12,20(13)             RECOVER OTHERS EXCEPT REG 115
         BR    14                      RETURN TO THE CALLER
ISGGREXE EQU   *                       END OF INSTRUCTIONS
         EJECT
***********************************************************************
*                                                                     *
* DATA USED                                                           *
*                                                                     *
***********************************************************************
         DS    OH
COMPRNME CLC   RNLERNME(ZERO),PELXRNME COMPARE RNAMES
*
```

```
COMPRNM1 CLC   RNLERNME(ZERO),UCBVOLI-UCBOB(R1) COMPARE VOLSER /*AAA
*OMPRNM1 CLC   RNLERNME(ZERO),X’1C’(R1)         COMPARE VOLSER /*AAA
*
EXCLNAME DC    CL8’RNLESEEX’           NAME IN END OF LIST ENTRY
*
HRDWNAME DC    CL8’HWRESERV’           MAJOR NAME PER ESCLUDE /*AAA
         SPACE 3
         PRINT NOGEN
         ISGPEL
         EJECT
         ISGRNLE
         IEFUCBOB DEVCLAS=DA
*
         END
```

Appendix C. MVS system commands

This appendix includes a table of MVS system commands and a table of MVS system commands that have a sysplex scope.

C.1 MVS system commands

Table 3 (Page 1 of 11). System command summary

Command (Abbr)	Function	Acceptable From	Command Group
ACTIVATE	Build the interface to and invoke the hardware configuration definition (HCD) application program interface.	MCS or extended MCS console (5)	SYS
CANCEL (C)	Cancel a MOUNT command Cancel a time-sharing user Cancel a cataloged procedure Cancel a job in execution Cancel a started catalog procedure Cancel an external writer allocation Cancel the writing of a SYSOUT data set by an external writer session Cancel a running APPC/MVS transaction program Cancel an OS/390 UNIX System Services process	MCS or extended MCS console or job stream (5)	SYS
CHNGDUMP (CD)	Override dump options specified in SYS1.PARMLIB, on the ABEND, CALLRTM, and SETRP macros, and in the SDUMP parameter list	MCS or extended MCS console or job stream (5)	SYS
CONFIG (CF)	Place processors online or offline Place central storage elements online or offline Place amounts of central storage online or offline Place ranges of central storage online or offline Place expanded storage elements online or offline Place channel paths online or offline Place Vector Facilities online or offline	MCS or extended MCS console (5)	MASTER

Table 3 (Page 2 of 11). System command summary

Command (Abbr)	**Function**	**Acceptable From**	**Command Group**
CONTROL (K)	Change display area specifications Delete certain messages Halt printing of a status display Control area displays Remove information from the screens Activate, deactivate, or display the status of the action message retention facility Change or display the number of allowed message and reply buffers Change or display message deletion or format specifications Change or display the status of WTO user exit IEAVMXIT Define commands for PFKs Reroute message queue Change or display time interval for updating a display Change operating mode of console Select the message levels for a console Increase the RMAX value In a sysplex, change the maximum time MVS waits before aggregating messages from routed commands	MCS console	INFO INFO INFO INFO INFO MASTER MASTER INFO MASTER INFO INFO or Master INFO INFO INFO MASTER MASTER
DEVSERV (DS)	Display current status of devices and corresponding channel paths	MCS or extended MCS console or job stream (5)	INFO

Table 3 (Page 3 of 11). System command summary

Command (Abbr)	**Function**	**Acceptable From**	**Command Group**
DISPLAY (D)	Display APPC/MVS configuration information Display ASCH configuration information Display IOS configuration Display console configuration information Display OS/390 UNIX System Services information Display MVS message service and current available languages Display status of external time reference (ETR) ports Display status information for trace Display system requests and status of the AMRF Display CONTROL command functions Display configuration information Display device allocation Display current system status Display system information requests Display local and Greenwich mean time and date Display domain descriptor table	MCS Console or extended MCS console or job stream (5)	INFO

Table 3 (Page 4 of 11). System command summary

Command (Abbr)	Function	Acceptable From	Command Group
DISPLAY (D)	Display APPC/MVS configuration information Display status or contents of SYS1.DUMP data sets and captured data sets Display dump options in effect Display SMF options in effect or SMF data sets Display information about the cross-system Coupling Facility information (XCF) Display information about operation information (OPDATA) in a sysplex Display information about the SMS configuration or the status of SMS volumes or storage groups or SMS trace options Display information about all subsystems defined to MVS. Display page and swap data set information Display current MIH time intervals for individual devices, or for device classes Display SLIP trap information Display commands defined for PFKs Display the messages MPF is processing and color, intensity, and highlighting display options in effect Display entries in the list of APF-authorized program libraries Display dynamic exits Display information about the LNKLST set Display information about modules dynamically added to LPA. Display state of the systems, a particular system′s CTCs, the status of an RNL change, or the contents of RNLs in the global resource serialization complex Display the active workload management service policy and mode status for systems or application environments Display information about registered products and the product enablement policy.	MCS Console or extended MCS console or job stream (5)	INFO
DUMP	Request a dump of virtual storage to be stored in a SYS1.DUMP data set	MCS or extended MCS console (5)	MASTER
DUMPDS (DD)	Change the system′s list of SYS1.DUMP data sets Clear full SYS1.DUMP data sets and make them available for dumps	MCS or extended MCS console (5)	SYS

Table 3 (Page 5 of 11). System command summary

Command (Abbr)	**Function**	**Acceptable From**	**Command Group**
FORCE	Force termination of: • A MOUNT command • A job in execution • An external writer allocation • The writing of a SYSOUT data set by an external writer • A non-cancellable job, time-sharing user, or started task • A running APPC/MVS transaction program	MCS or extended MCS console (5)	MASTER
HALT (Z)	Record statistics before stopping the system (must first stop subsystem processing with a subsystem command)	MCS or extended MCS console (5)	SYS
IOACTION (IO)	Stop or resume I/O activity to DASD	MCS or MASTER extended MCS console (5)	MASTER
LIBRARY (LI)	Eject a volume from a library of removable storage media Reactivate processing for certain installation exits without stopping or restarting object access method (OAM) Set or display the media type of scratch volumes that the system places into the cartridge loader of a device within a tape library Display tape drive status	MCS console	SYS
LOG (L)	Enter comments in the system log	MCS or extended MCS console or job stream (5)	INFO
LOGOFF	Log off MCS consoles	MCS console	SYS
LOGON	Access the MCS consoles	MCS console	SYS
MODE	Control recording of or suppress system recovery and degradation machine check interruptions on the logrec data set Control the monitoring of hard machine check interruptions	MCS or extended MCS console (5)	SYS

Table 3 (Page 6 of 11). System command summary

Command (Abbr)	**Function**	**Acceptable From**	**Command Group**
MODIFY (F)	Change characteristics of a job by modifying the job parameters Specify criteria an external writer uses to select data sets for processing Cause an external writer to pause for operator intervention Start TSO/TCAM time-sharing once TCAM is active Stop TSO/TCAM time-sharing Build a new LLA directory Display information about the catalog address space or request the catalog address space to perform a specified service Modify TSO/VTAM time-sharing Rebuild a new LNKLST directory Display the status of the DLF, or change DLF parameters or processing mode Switch the workload management mode in effect for a system	MCS or extended	SYS
MONITOR (MN)	Continuously display data set status Continuously display job status Monitor time-sharing users logging on and off the system	MCS or extended MCS console or job stream (5)	INFO
MOUNT (M)	Mount volumes	MCS or extended MCS console or job stream (5)	I/O
MSGRT (MR)	Establish message routing instructions for certain options of DISPLAY, TRACK, STOPTR, MONITOR, STOPMN, CONFIG, and CONTROL commands Stop message routing Establish message routing for OS/390 UNIX System Services information	MCS Console	INFO
PAGEADD (PA)	Add local page or swap data sets Specify data sets as non-VIO page data sets	MCS or extended MCS console (5)	SYS
PAGEDEL (PD)	Delete, replace, or drain a local page data set or swap data set PLPA, common, duplex page data sets, and the last local page data set cannot be deleted, replaced, or drained	MCS or extended MCS console (5)	SYS
QUIESCE	Put system in MANUAL state without affecting step timing	MCS or extended MCS console (5)	MASTER

Table 3 (Page 7 of 11). System command summary

Command (Abbr)	Function	Acceptable From	Command Group
REPLY (R)	Reply to system information requests Reply to system requests during recovery processing Specify component trace options after issuing TRACE CT Specify system parameters Set the time-of-day clock and specify the installation performance specification Specify SMF options Specify DUMP options	MCS or extended MCS console or job stream (5)	INFO
RESET (E)	Change performance group of a job currently in execution Assign work to a new workload management service class. Also, quiesce and resume executing work. Force a hung console device offline.	MCS or extended MCS console or job stream (5)	SYS
ROUTE (RO)	Direct a command to another system, to all systems, or to a subset of systems in the sysplex	MCS or extended MCS console or job stream (5)	INFO
SEND (SE)	Communicate with other operators Communicate with specific time-sharing users Communicate with all time-sharing users Save messages in the broadcast data set for issuance at TSO LOGON time or when requested List messages accumulated in the notices section of the broadcast data set Delete a message from the notices section of the broadcast data set	MCS or extended MCS console or job stream (5)	INFO

Table 3 (Page 8 of 11). System command summary

Command (Abbr)	Function	Acceptable From	Command Group
SET (T)	Change local time and date Change system resources manager (SRM) parameters Change MPF parameters Restart SMF or change SMF parameters by changing the active SMFPRMxx member of SYS1.PARMLIB Change the dump analysis and elimination (DAE) parameters Change SLIP processing by changing the active IEASLPxx member of SYS1.PARMLIB Change SMS parameters by selecting member IGDSMSxx in SYS1.PARMLIB, start SMS if not started at IPL, or restart SMS if it can′t be automatically restarted Change available PFK tables Change MIH time intervals by changing the active IECSIOSxx member of SYS1.PARMLIB Change excessive spin-loop timeout interval recovery actions Change RNLs by selecting new members of GRSRNLxx in SYS1.PARMLIB Change the APPC/MVS address space information Change the APPC/MVS transaction scheduler information Change the PPT information Change the active console group definitions in the sysplex Start, refresh, or stop MMS Update the format or contents of the APF list Start or stop the common storage and tracking functions Change MMS parameters Start, stop, or refresh MMS Update APF list and dynamic exits Update the LNLKST set for the LNKLST concatenation Dynamically add modules to, or delete modules from, the LPA Change the command installation exits the system is to use Change the product enablement policy the system is to use	MCS or extended MCS console or job stream (5)	SYS
SETDMN (SD)	Respecify domain parameters	MCS or extended MCS console or job stream (5)	SYS

Table 3 (Page 9 of 11). System command summary

Command (Abbr)	**Function**	**Acceptable From**	**Command Group**
SETETR	Enable external time reference (ETR) ports that have been disabled	MCS or extended MCS console (5)	SYS
SETGRS	Migrate a currently active global resource serialization ring complex to a global resource serialization star complex	MCS or extended MCS console (5)	SYS
SETIOS	Respecify, add, or delete MIH time intervals without changing the active IECIOSxx member of SYS1.PARMLIB	MCS or extended MCS console (5)	SYS
SETLOAD	Switch dynamically from one parmlib concatenation to another without having to initiate an IPL	MCS or extended MCS console (5)	SYS
SETLOGRC	Change the logrec recording medium	MCS or extended MCS console (5)	SYS
SETOMVS	Change the options that OS/390 UNIX System Services uses	MCS or extended MCS console (5)	SYS
SETPROG	Update APF list Update dynamic exits Update the LNKLST set Dynamically add modules to, or delete modules from, LPA	MCS or extended MCS console (5)	SYS
SETRRS CANCEL	End RRS processing	MCS or extended MCS console (5)	SYS
SETSMF (SS)	Change SMF parameters without changing the active SMFPRMxx member of SYS1.PARMLIB	MCS or extended MCS console or job stream (5)	SYS
SETSMS	Change SMS parameters without changing the active IGDSMSxx member of SYS1.PARMLIB	MCS or extended MCS console (5)	SYS
SETSSI	Dynamically add, activate, or deactivate a subsystem	MCS or extended MCS console (5)	SYS
SETXCF	Control the cross-system Coupling Facility (XCF)	MCS or extended MCS console (5)	MASTER
SLIP (SL)	Set SLIP traps Modify SLIP traps Delete SLIP traps	MCS or extended MCS console or job stream (5)	SYS

Table 3 (Page 10 of 11). System command summary

Command (Abbr)	**Function**	**Acceptable From**	**Command Group**
START (S)	Start a job from a console Start the advanced program-to-program communication (APPC/MVS) address space Start the APPC/MVS scheduler (ASCH) address space Start the data facility storage management subsystem (DFSMS/MVS) license compliance facility Start the generalized trace facility (GTF) Start the library lookaside (LLA) address space Start the object access method (OAM) Start resource recovery services (RRS) Start the system object model (SOM) Start TSO/VTAM time-sharing Start the virtual lookaside facility (VLF) or the data lookaside facility (DLF) Start an external writer	MCS or extended MCS console or job stream (5)	SYS
STOP (P)	Stop a job in execution Stop an address space Stop an ASCH initiator Stop an initiator Stop the data lookaside facility (DLF) Stop the generalized trace facility (GTF) Stop the library lookaside (LLA) address space Stop the object access method (OAM) Stop the system object model (SOM) Stop TSO/VTAM time-sharing Stop the virtual lookaside facility (VLF) Stop an external writer	MCS or extended MCS console or job stream (5)	SYS
STOPMN (PM)	Stop continual display of data set status Stop continual display of job status Stop monitoring the activity of time-sharing users.	MCS or extended MCS console or job stream (5)	INFO
STOPTR (PT)	Halt or reduce information displayed with the TRACK command	MCS Console	INFO
SWAP (G)	Move a volume from one device to another	MCS or extended MCS console (5)	I/O
SWITCH (I)	Manually switch recording of SMF data from one data set to another Switch a consoles attributes to another console	MCS or extended MCS console (5)	SYS MASTER

Table 3 (Page 11 of 11). System command summary

Command (Abbr)	**Function**	**Acceptable From**	**Command Group**
TRACE	Start, stop, or modify system trace Start, stop, or modify master trace Start, stop, or modify component trace Display the status of system trace, master trace, or component trace	SYS MASTER MASTER SYS	
TRACK (TR)	Periodically display job information on display consoles Periodically display unit of work information executing for APPC/MVS transaction requestor Periodically display the number of OpenMVS address spaces	MCS Console	INFO
UNLOAD (U)	Remove a volume from system use	MCS or extended MCS console or job stream (5)	I/O
VARY (V)	Change the master console Control the hardcopy message set and the hardcopy medium. Change the status of a secondary console Change the console′s alternate console group Change the SMS status of a storage group or volume for one or more MVS systems in the SMS complex Place I/O devices online or offline Assign and control consoles Place I/O paths online or offline Remove a system from a sysplex Place I/O paths online after C.U.I.R service Change a system′s participation in a global resource serialization complex Change routing codes for a console Activate a workload management service policy for a sysplex Control an application environment	MCS console MCS console or job stream MCS console	(Note 4) MASTER (Note 4) MASTER
WRITELOG (W)	Schedule printing of system log Change system log output class Close system log and discontinue log function Restart system log after closing	MCS or extended MCS console or job stream (5)	SYS

Notes:

1. CONS command group when message routing is specified.
2. For information about VTAM commands, see VTAM Operation.
3. For information about TCAM commands, see ACF/TCAM Operation.
4. I/O command group when specifying a non-console device; CONS when specifying

5. An extended MCS console can be either an interactive TSO/E session or a prog MCSOPER macro.

C.2 MVS system commands with a sysplex scope

Table 4 (Page 1 of 3). MVS system commands with sysplex scope

Command	Conditions
CHNGDUMP	Has sysplex scope only when all systems are connected to the same Coupling Facilities, and you specify ,SDUMP,SYSFAIL,STRLIST=.
CONTROL C,A	All
CONTROL C,D	Has sysplex scope only when you specify L=.
CONTROL M	Has sysplex scope only when you do not specify MLIM, UEXIT, or LOGLIM.
CONTROL other	Other parameters of CONTROL have sysplex scope only when you specify L=.
DISPLAY CF	Has sysplex scope only when displaying for those systems connected to the Coupling Facility. Does not have sysplex scope when displaying an individual system′s Coupling Facility configuration information (Coupling Facility channels and paths).
DISPLAY CNGRP	All
DISPLAY CONSOLES	Has sysplex scope unless you specify DISPLAY C,B or DISPLAY C,U.
DISPLAY DUMP	Has sysplex scope only when you issue the OPTIONS parameter to display the results of a CHNGDUMP ...SDUMP,SYSFAIL,STRLIST= command.
DISPLAY GRS	Has sysplex scope unless you specify SUSPEND. Also, note the following about DISPLAY GRS,C and DISPLAY GRS,RES: the output generated by these commands includes both system-specific information (S=SYSTEM) and sysplex information (S=SYSTEMS). The S=SYSTEM information is valid only for the system on which you issue the command. The S=SYSTEMS information is identical regardless of the system on which you issue the command.
DISPLAY OPDATA	All
DISPLAY PFK	Has sysplex scope only when you specify CN=.
DISPLAY R	Has sysplex scope, but the output might be different on different consoles, because the output of DISPLAY R is dependent on the routing criteria for the console specified by CN=. If you do not specify CN=, the routing criteria of the console issuing the command is used. If you issue the command in a program (by using the MGCRE macro) the console you specify in the macro is used. If you specify a console ID of 0, all retained messages are included in the command response.
DISPLAY WLM	All
DISPLAY XCF,ARMSTATUS	Has sysplex scope provided all systems are using the same ARM couple data set.
DISPLAY XCF,CF	Has sysplex scope provided all systems in the sysplex are connected to the same coupling facilities.
DISPLAY XCF,COUPLE	Has sysplex scope as long as all systems are using the same types of couple data sets, as specified on the TYPE parameter (SYSPLEX, ARM, CFRM, SFM, LOGR, and WLM.) If you do not specify the TYPE parameter, only system-specific data is displayed.
DISPLAY XCF,GROUP	All
DISPLAY XCF,POLICY	Has sysplex scope as long as all systems are using the same types of couple data sets, as specified on the TYPE parameter (ARM, CFRM, SFM, and LOGR.)
DISPLAY XCF,STRUCTURE	Has sysplex scope provided all systems in the sysplex are connected to the same coupling facilities.
DISPLAY XCF,SYSPLEX	All
MONITOR	Has sysplex scope only when you specify L=.

Table 4 (Page 2 of 3). MVS system commands with sysplex scope

Command	Conditions
MOUNT	Has sysplex scope only when you issue the command against an automatically switchable tape device.
REPLY	All
RESET CN	Issue the command from the system where the console was attached to avoid inconsistent sysplex results.
SEND	Has sysplex scope only when sending to consoles; does not have sysplex scope when sending to TSO users.
SET CNGRP	Has sysplex scope provided all systems are sharing the same SYS1.PARMLIB data set.
SET DAE	Has sysplex scope only when all systems are sharing the same DAE data set, and the same SYS1.PARMLIB data set.
SET GRSRNL	Has sysplex scope only when all systems are sharing the same SYS1.PARMLIB data set.
SET SMS	Has sysplex scope when you are issuing the command to change the name of the ACDS or COMMDS. All systems in the sysplex must be in the same SMS complex, and using the same SYS1.PARMLIB data set. If you are issuing the command to start or restart SMS on a system, only the system on which you issue the command is affected.
SETSMS	Has sysplex scope only if you are changing the SCDS, ACDS, or COMMDS, and only if all systems in the sysplex are in the same SMS complex.
SETXCF FORCE	Has sysplex scope only when all systems are connected to the same Coupling Facility.
SETXCF COUPLE	Has sysplex scope only when you specify PSWITCH, ACOUPLE, or PCOUPLE, and all systems have access to the specified couple data set.
SETXCF START]STOP	Have sysplex scope only when you specify POLICY or REBUILD.
STOPMN	Has sysplex scope only when you specify L=.
STOPTR	Has sysplex scope only when you specify L=.
SWITCH CN	All
TRACK	Has sysplex scope only when you specify L=.
UNLOAD	Has sysplex scope only when you issue the command against an automatically switchable tape device.
VARY CN	Has sysplex scope unless all of the following are true: • You issue VARY CN(conspec),ONLINE without specifying SYSTEM=. • You do not specify SYSTEM= in the CONSOLxx member of SYS1.PARMLIB that defines this console. • The console has never been active in the sysplex.
VARY ...,MSTCONS	Has sysplex scope when you issue VARY conname,MSTCONS. Also has sysplex scope when you issue VARY devnum,MSTCONS, but only if you use a common IODF for the specified device across the sysplex.
VARY SMS, STORGRP]VOLUME	Has sysplex scope under these conditions only: • You specify (storgrp]volume,ALL) and all systems in the sysplex are in the same SMS complex. • You specify (storgrp]volume,system) where system is a system group, and the system group exactly matches the sysplex (that is, none of the systems in the sysplex is explicitly defined to SMS).
VARY XCF	All

Table 4 (Page 3 of 3). MVS system commands with sysplex scope

Command	**Conditions**
VARY WLM	All

Appendix D. OS/390 problem diagnosis

This appendix provides the following information:

- ADMSADMP macro parameters
- VTAM internal trace options

D.1 ADMSADMP macro parameters

The ADMSADMP macro parameter definitions are:

```
symbol  AMDSADMP
IPL={Tunit|Dunit|DSYSDA}
VOLSER={volser|SADUMP}
ULABEL={PURGE|NOPURGE}
CONSOLE=({cnum|(cnum,ctype),(cnum,ctype) ...|01F,3278})
SYSUT={unit|SYSDA}
OUTPUT={Tunit|Dunit|(Dunit,ddsname)|T0282}
DUMP=′options′   ,PROMPT
MSG={ACTION|ALLASIDS|ALL}
MINASID={ALL|PHYSIN}
COMPACT={YES|NO}
REUSEDS={CHOICE|ALWAYS|NEVER}
DDSPROMT={YES|NO}
```

Where:

symbol An arbitrary name you can assign to the AMDSADMP macro. SADMP uses this symbol to create a job name for use in the initialization step. AMDSADMP is the name of the macro.

IPL {Tunit|Dunit|DSYSDA} indicates the device number, device type, or esoteric name of the stand-alone dump residence volume. The first character indicates the volume type; T for tape, D for DASD. SADMP uses the unit character string as the UNIT=value to allocate the residence volume for initialization.

A device number consists of one to four hexadecimal digits, optionally preceded by a slash (/). Use a slash preceding a four-digit device number to distinguish it from a device type.

Note: This device will also contain a work file used during stand-alone dump processing.

VOLSER {volser|SADUMP} indicates the volume serial number the system is to use to allocate the residence volume for initialization. When you specify a tape volume, it must be NL (no labels). VOLSER=SADUMP is the default.

ULABEL {PURGE|NOPURGE} indicates whether SADMP deletes (PURGE) or retains (NOPURGE) existing user labels on a DASD residence volume. When you specify NOPURGE, the SADMP program is written on cylinder 0 track 0 of the residence volume, immediately following all user labels. If the user labels occupy so much space that the SADMP program does not fit on track 0, the initialization program issues an error message and ends.

CONSOLE ({cnum|(cnum,ctype),(cnum,ctype)..|01F,3278}) indicates the device numbers and device types of the SADMP consoles that SADMP is to use while taking the dump. When you specify CONSOLE=cnum, SADMP assumes (cnum,3278). You can specify from two to 21 consoles by coding:

```
CONSOLE=((cnum,ctype),(cnum,ctype),      ,(cnum,ctype) ...)
```

A device number consists of three or four hexadecimal digits, optionally preceded by a slash (/). Use a slash preceding a four-digit device number to distinguish it from a device type.

The 3277, 3278, 3279, and 3290 device types are valid, and are interchangeable.

Note: The specification of CONSOLE does not affect the availability of the system console.

SYSUT {unit|SYSDA} specifies the UNIT=value of the device that SADMP uses for work files during stand-alone dump initialization. You may specify the device as a group name (for example, SYSDA), a device type (for example, 3330), or a unit address (for example, 131). SYSUT=SYSDA is the default.

OUTPUT {Tunit|Dunit|(Dunit,ddsname)| T0282} indicates the device type, number, and data set name that SADMP uses as a default value if the operator uses the EXTERNAL INTERRUPT key bypass console communication, or if the operator provides a null response to message AMD001A during SADMP initialization. OUTPUT=T0282 is the default.

The device type can be specified as either a ′T′ for tape or ′D′ for DASD.

The device number consists of three or four hexadecimal digits, optional preceded by a slash (/). Use a slash preceding a four-digit device to distinguish it from a device type.

If the default device is a DASD, you can also set up a default dump data set name to use by specifying both the device and the dump data set name on the OUTPUT= parameter. If you specify a default dump data set name it must:

- Have a length that is 44 characters or less.
- Contain the text ′SADMP′ as either part of, or as an entire set qualifier.

DUMP ′options′ indicates additional virtual storage that you want dumped. This storage is described as address ranges, dataspaces, and subpools address spaces. When you do not specify DUMP, SADMP does not dump additional storage unless you specify PROMPT.

PROMPT causes SADMP, at run time, to prompt the operator for additional virtual storage to be dumped. The operator can respond with the same information that can be specified for the DUMP keyword. When you do not specify PROMPT, SADMP does not prompt the operator to specify additional storage.

MSG {ACTION|ALLASIDS|ALL} indicates the type of SADMP messages that appear on the console. When you specify ACTION, SADMP writes only messages that require operator action. When you specify ALL, SADMP writes most messages to the console. However, messages AMD010I, AMD057I, AMD076I, AMD081I, and AMD102I appear only in the SADMP message log. When you specify ALLASIDS, the SADMP program behaves as if MSG=ALL was specified, except that message AMD010I also appears on the console. ALL is the default.

This keyword has no effect on the SADMP message log; even if you specify MSG=ACTION, the SADMP virtual dump program writes all messages to the message log in the dump.

This keyword has no effect on the SADMP message log; even if you specify MSG=ACTION, the SADMP virtual dump program writes all messages to the message log in the dump.

MINASID {ALL|PHYSIN} indicates the status of the address spaces that are to be included the minimal dump. Specify PHYSIN to dump the minimum virtual storage (LSQA and selected system subpools) for the physically swapped-in address spaces only. Specify

ALL to dump the minimum virtual storage (LSQA and selected system subpools) for all of the address spaces. ALL is the default.

At run time, if PHYSIN was specified, SADMP writes message AMD082I to the operator′s console to warn the operator that some virtual storage might be excluded from the dump.

COMPACT COMPACT(YES) compacts the data stored on a tape cartridge if the IDRC hardware feature is available on your tape drive. If the IDRC feature is available and you do not specify the COMPACT keyword, the default is YES, so that IDRC will compact the dump data. Otherwise, the data is handled as usual.

REUSEDS { CHOICE|ALWAYS|NEVER} indicates whether SADMP should reuse the dump data set on the specified output device when it determines that the data set is valid, however, it may contain data from a previous dump. SADMP determined this by checking to see if the first record in the data set matches the record that is written by the AMDSADDD REXX utility. When you specify ALWAYS, SADMP issues message AMD094I and reuses the specified dump data set. When you specify NEVER, SADMP issues message AMD093I and prompts the operator, through message AMD001A, for an output device. When you specify CHOICE, SADMP informs the operator, with message AMD096A, that the data set is not reinitialized and requires permission to reuse the data set.

DDSPROMPT DDSPROMPT=YES allows the stand-alone dump program to prompt the operator for an output dump data set when dumping to a DASD device. When DDSPROMPT=YES is specified, after replying to message AMD001A with a DASD device number, message AMD002A will also be issued to prompt the operator for a dump data set name.

DDSPROMPT=NO indicates that the stand-alone dump program should not prompt for a dump data set name when dumping to a DASD device. When DDSPROMPT=NO is specified, after replying to message AMD001A with a DASD device number, the SADMP program will assume that the data set SYS1.SADMP is to be used. DDSPROMPT=NO is the default.

Note: Regardless of the DDSPROMPT= keyword value, you can always use a default device and dump data set name by specifying the OUTPUT=(Dunit,ddsname) keyword. The SADMP program uses the default values specified on the OUTPUT= keyword when the operator uses the External Interrupt key to bypass console communication, or if the operator provides a null response to message AMD001A.

D.2 VTAM internal trace options

Both the TRACE start option and the `MODIFY TRACE` command have an OPTION operand you can use to select VIT options. Select one or more of the following options to indicate the VTAM functions you want to trace.

Note: If you do not deactivate the VIT before you attempt to change an option, the options that are currently in effect will remain in effect.

The options are as follows:

API API option (Application Program Interfaces) - This option helps you determine whether an application program is causing a problem. API entries are written for RPL macros, RPL exit routines, user exit routines, and user posts. Trace data for this option is always automatically recorded in the internal table.

APPPC APPC option (LU 6.2 Application Program Interfaces) - This option helps you determine whether an LU 6.2 application is causing a problem. LU 6.2 entries are written for APPCCMD macro invocations, user posts and exit scheduling by LU 6.2 code, calls to a security manager for security processing, and message unit transmissions between LU 6.2 components.

CFS CFS option (for coupling facility interfaces) - This option helps you determine problems with VTAM′s interface with the MVS coupling facility. CFS entries are written when VTAM issues MVS macros to request coupling facility related services.

CIO CIO option (for channel input and output) - This option helps you isolate problems related to channel I/O. CIO entries are written for attentions, error recovery, interruptions, HALT I/O SVC, and START I/O SVC.

CMIP CMIP option (common management information protocol services) - Setting the CMIP option traces:

- Calls from CMIP application programs to the management information base (MIB) application programming interface
- Calls to the read-queue exit of the CMIP application program
- Topology updates from VTAM resources

CSM CSM option (communications storage manager events) - This option traces the parameter list information that flows across the CSM interface and key internal events (such as pool expansion and contraction) for functions that manipulate buffer states. This allows you to trace and analyze the usage history of a buffer. You can also use the CSM trace when VTAM is not operational. An external trace is generated using the VTAM GTF event ID to write trace records directly to GTF in the same format as those recorded using VIT.

ESC ESC option (execution sequence control) - This option helps you track in detail the flow of requests for a given process.

HPR HPR option (high performance routing) - This option helps you isolate problems related to high performance routing.

LCS LCS option (local area network (LAN) channel stations) - This option helps you isolate problems occurring during activation of, deactivation of, and data transfer from an IBM 3172 Interconnect Nways Controller. The LCS option enables tracing of data that VTAM receives from an IBM 3172 Interconnect Nways Controller. If the VIT is active and VTAM receives a frame that is not valid from an IBM 3172 Interconnect Nways Controller, an LCSL, LCSP, or LCSS trace record is created depending on the error conditions.

LOCK LOCK option (Locking and Unlocking) This option helps you determine when VTAM modules get and release locks.

MSG MSG option (messages) - This option helps you:

- Correlate other VIT entries with the console messages even if you lose the console sheet. MSG entries are written for all messages to the VTAM operator.
- Match the console log to a surge of activity shown in the VIT. OPER entries are written for all VTAM commands issued at an operator console.

Trace data for this option is always automatically recorded in the internal table.

NRM NRM option (network resource management) - This option helps you follow the services of the network resource management component. These include the assignment of, references to, and the deletion of certain VTAM resources such as node names, network addresses, and control blocks. NRM entries are written for SRT macros issued by VTAM modules. Trace data for this option is always automatically recorded in the internal table.

PIU PIU option (path information unit flows) - This option, like the I/O and buffer contents traces, helps you isolate problems to hardware, to the NCP, or to VTAM. Unlike I/O and buffer contents traces, with this option PIU entries are written for all PIUs that flow internal and external to VTAM. Trace data for this option is always automatically recorded in the internal table.

PSS PSS option (process scheduling services) - This option helps you track the flow of requests through VTAM. PSS entries are written for the VTAM macros that invoke and control PSS, scheduling and dispatching VTAM routines.

SMS SMS option (storage management services) - This option helps you isolate problems caused by storage shortages. When used with the SSCP or PSS trace options, it can also help you isolate internal VTAM problems. SMS entries are written when SMS macros are used to request or free fixed- or variable-length buffers. SMS entries are also written when VTAM expands or attempts to expand a buffer pool.

SSCP SSCP option (system services control point request) - This option helps you isolate a VTAM problem to a specific VTAM component or module. SSCP entries are written for the request/response units (RUs) sent between VTAM components. This option also records information for the APPN CP. Trace data for this option is always automatically recorded in the internal table.

TCP TCP option (for use with the SNA over TCP/IP feature application program interfaces) - This option helps you trace the communication between VTAM and TCP/IP. Events include:

- Socket API calls and completions
- IUCV calls and completions
- Invocations of the SOCTREE macro, which controls updating a binary tree of socket control blocks
- Tracing MPTN formats

VCNS VCNS option (VCNS application program interfaces) - This option helps you determine whether a VCNS application is causing a problem. VCNS entries are written for VCNSCMD macro invocations, user posts, and exit scheduling by VCNS code, and work element transmissions between VCNS components.

XBUF XBUF option (for applications using the extended buffer list for sending and receiving data) - This option traces the contents of the extended buffer list (XBUFLST). Records are produced to trace these contents from the application-supplied extended buffer list as well as the internal buffer list that VTAM uses to carry the extended buffer list information. These records store relevant information contained with the extended buffer list, particularly information on CSM usage by VTAM.

XCF XCF option (for VTAM′s use of the cross-system coupling facility) - This option allows you to track VTAM′s use of the XCF (cross-system coupling facility) MVS macro interface. There is a VIT entry for each VTAM use of an XCF macro.

Appendix E. Special Notices

This publication is intended to help new system programmers who need to understand S/390 and the OS/390 operating system. The information in this publication is not intended as the specification of any programming interfaces that are provided by OS/390 Versions. See the PUBLICATIONS section of the IBM Programming Announcement for OS/390 Version 2 Release 8, Program Number 5647-A01 for more information about what publications are considered to be product documentation.

References in this publication to IBM products, programs or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM′s product, program, or service may be used. Any functionally equivalent program that does not infringe any of IBM′s intellectual property rights may be used instead of the IBM product, program or service.

Information in this book was developed in conjunction with use of the equipment specified, and is limited in application to those specific hardware and software products and levels.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM Corporation, Dept. 600A, Mail Drop 1329, Somers, NY 10589 USA.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

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

Any pointers in this publication to external Web sites are provided for convenience only and do not in any manner serve as an endorsement of these Web sites.

Reference to PTF numbers that have not been released through the normal distribution process does not imply general availability. The purpose of including these reference numbers is to alert IBM customers to specific information relative to the implementation of the PTF when it becomes available to each customer according to the normal IBM PTF distribution process.

You can reproduce a page in this document as a transparency, if that page has the copyright notice on it. The copyright notice must appear on each page being reproduced.

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

ACF/VTAM
Advanced Function Printing
AFP
AnyNet
CICS
CICS/ESA
CICS/MVS
DB2

DFSMS
DFSMS/MVS
DFSMSdfp
DFSMSdss
DFSMShsm
DFSMSrmm
DFSORT
ESCON
FICON
IBM
IMS
InfoPrint
Intelligent Printer Data Stream
IPDS
IP PrintWay
Language Environment
MVS (block letters)
MVS/DFP
NetSpool
NetView
OpenEdition
OS/390
Parallel Sysplex
PrintWay
RACF
RAMAC
RMF
S/390
S/390 Parallel Enterprise Server
Sysplex Timer
VTAM

The following terms are trademarks of other companies:

The following terms are trademarks of other companies:

Tivoli, Manage. Anything. Anywhere., The Power To Manage., Anything. Anywhere., TME, NetView, Cross-Site, Tivoli Ready, Tivoli Certified, Planet Tivoli, and Tivoli Enterprise are trademarks or registered trademarks of Tivoli Systems Inc., an IBM company, in the United States, other countries, or both. In Denmark, Tivoli is a trademark licensed from Kjobenhavns Sommer - Tivoli A/S.

C-bus is a trademark of Corollary, Inc. in the United States and/or other countries.

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

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

PC Direct is a trademark of Ziff Communications Company in the United States and is used by IBM Corporation under license.

ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States and/or other countries.

SET, SET Secure Electronic Transaction, and the SET logo are trademarks owned by Secure Electronic Transaction LLC.

UNIX is a registered trademark in the United States and other countries licensed exclusively through the Open Group.

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

Appendix F. Related Publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

F.1 IBM Redbooks

For information on ordering these ITSO publications see “How to get IBM Redbooks” on page 389.

- *OS/390 Release 5 Implementation*, SG24-5151
- *OS/390 Release 4 Implementation*, SG24-2089
- *OS/390 Release 3 Implementation*, SG24-2067
- *OS/390 Release 2 Implementation*, SG24-4834
- cit.OS/390 Security Server 1999 Updates Technical Presentation Guide, SG24-5627
- *Security in OS/390-based TCP/IP Networks*, SG24-5383
- *Hierarchical File System Usage Guide*, SG24-5482
- *Enhanced Catalog Sharing and Management*, SG245594
- *Integrated Catalog Facility Backup and Recovery*, SG24-5644
- *OS/390 Version 2 Release 6 UNIX System Services Implementation and Customiztion*, SG24-5178
- *IBM S/390 FICON Implementation Guide*, SG24-5169
- *Exploiting S/390 Hardware Cryptography with Trusted Key Entry*, SG24-5455
- *TCP/IP Tutorial and Technical Overview*, GG24-3376
- *Introduction to Storage Area Network SAN*, SG2-45470
- *TCP/IP in a Sysplex*, SG24-5235
- *SecureWay Communications Server for OS/390 V2R8 TCP/IP: Guide to Enhancements*, SG24-5631
- *OS/390 eNetwork Communications Server V2R7 TCP/IP Implementation Guide Volume 1: Configuration and Routing*, SG24-5227
- *OS/390 eNetwork Communications Server V2R7 TCP/IP Implementation Guide Volume 3: MVS Applications*, SG24-5229
- *OS/390 Workload Manager Implementation and Exploitation*, SG24-5326
- *ADSM Server-to-Server Implementation and Operation*, SG24-5244
- *Stay Cool on OS/390: Installing Firewall Technology*, SG24-2046
- *Implementing DFSMSdss SnapShot and Virtual Concurrent Copy*, SG24-5268
- *TCP/IP OpenEdition Implementation Guide*, SG24-2141
- *IMS/ESA Version 5 Performance Guide*, SG24-4637
- *Parallel Sysplex Configuration: Overview*, SG24-2075
- *Parallel Sysplex Configuation: Cookbook*, SG24-2076
- *Parallel Sysplex Config.: Connectivity*, SG24-2077
- *DFSMS Optimizer Presentation Guide Update*, SG24-4477
- *MVS Parallel Sysplex Capacity Planning*, SG24-4680

- *Getting the Most Out of a Parallel Sysplex*, SG24-2073
- *OS/390 eNetwork Communication Server TCP/IP Implementation Guide Volume 2*, SG24-5228

F.2 IBM Redbooks collections

Redbooks are also available on the following CD-ROMs. Click the CD-ROMs button at `http://www.redbooks.ibm.com/` for information about all the CD-ROMs offered, updates and formats.

CD-ROM Title	**Collection Kit Number**
System/390 Redbooks Collection	SK2T-2177
Networking and Systems Management Redbooks Collection	SK2T-6022
Transaction Processing and Data Management Redbooks Collection	SK2T-8038
Lotus Redbooks Collection	SK2T-8039
Tivoli Redbooks Collection	SK2T-8044
AS/400 Redbooks Collection	SK2T-2849
Netfinity Hardware and Software Redbooks Collection	SK2T-8046
RS/6000 Redbooks Collection (BkMgr Format)	SK2T-8040
RS/6000 Redbooks Collection (PDF Format)	SK2T-8043
Application Development Redbooks Collection	SK2T-8037
IBM Enterprise Storage and Systems Management Solutions	SK3T-3694

F.3 Other resources

These publications are also relevant as further information sources:

- *OS/390 Initialization and Tuning Guide*, SC28-1751
- *OS/390 Initialization and Tuning Reference*, SC28-1752
- *OS/390 Introduction and Release Guide*, GC28-1725
- *OS/390 MVS JCL User′s Guide*, SC28-1758
- *OS/390 MVS JCL Reference*, GC28-1757
- *OS/390 MVS System Diagnosis: Tools and Service Aids*, LY28-1085, (available to IBM licensed customers only)
- *Interactive System Productivity Facility Getting Started*, SC34-4440
- *OS/390 Security Server (RACF) System Programmer′s Guide*, SC28-1913
- *OS/390 TSO/E Customization*, SC28-1965
- *OS/390 TSO/E Primer*, GC28-1967
- *OS/390 TSO/E User′s Guide*, SC28-1968
- *OS/390 SMP/E Reference*, SC28-1806
- *OS/390 SMP/E User′s Guide*, SC28-1740
- *OS/390 SMP/E Commands*, SC28-1805
- *Standard Packaging Rules for MVS-Based Products*, SC23-3695
- *OS/390 MVS System Commands*, GC28-1781
- *OS/390 MVS IPCS Commands*, GC28-1754
- *OS/390 MVS IPCS User′s Guide*, GC28-1756
- *DFSMS/MVS Using Data Sets*, SC26-4922
- *OS/390 Planning for Installation*, GC28-1726
- *OS/390 MVS System Data Sets Definition*, GC28-1782

- *ICKDSF R16 Refresh, User's Guide*, GC35-0033
- *OS/390 MVS System Management Facilities (SMF)*, GC28-1783
- *EREP V3R5 Reference*, GC35-0152
- *OS/390 JES2 Commands*, GC28-1790
- *OS/390 Hardware Configuration Definition User's Guide*, SC28-1848
- *DFSMS/MVS DFSMSdss Storage Administration Reference*, SC26-4929
- *IBM ServerPac for OS/390 Using the Installation Dialog*, SC28-1244
- *OS/390 Hardware Configuration Definition Planning*, GC28-1750
- *OS/390 MVS Using the Subsystem Interface*, SC28-1502
- *DFSMS/MVS Version 1 Release 4: Managing Catalogs*, SC26-4914
- *DFSMS/MVS Version 1 Release4: Access Method Services for Integrated Catalog Facility*, SC26-4906
- *DFSMS/MVS: DFSMShsm Implementation and Customization Guide*, SH21-1078
- *DFSMS/MVS Access Method Services for ICF Catalogs*, SC26-4500
- *DFSMS/MVS DFSMSdfp Storage Administration Reference*, SC26-4920
- *OS/390 eNetwork Communications Server: SNA Resource Definition Reference*, SC31-8565
- *OS/390 eNetwork Communications Server SNA Resource Definition Samples*, SC31-8566
- *OS/390 eNetwork Communications Server: SNA Operation*, SC31-8567
- *OS/390 V2R7.0 eNetwork CS IP Configuration*, SC31-8513
- *eNetwork Communications Server: IP User's Guide* GC31-8514
- *OS/390 UNIX System Services Planning*, SC28-1890
- *OS/390 TCP/IP OpenEdition: Configuration Guide* SC31-8304
- *OS/390 Open Systems Adapter Support Facility User's Guide*, SC28-1855.
- *OS/390 V2R6.0 MVS Planning: APPC/MVS Management*, GC28-1807
- *Print Services Facility for OS/390: Customization*, S544-5622
- *DFSMS/MVS Planning for Installation*, SC26-4919
- *DFSMS/MVS Implementing System-Managed Storage*, SC26-3123
- *DFSMS/MVS Remote Copy Administrator's Guide and Reference*, SC35-0169
- *DFSMS/MVS Macro Instructions for Data Sets*, SC26-4913
- *DFSMS/MVS DFSMSdfp Diagnosis Guide*, SY27-9605
- *DFSMS/MVS DFSMSdfp Advanced Services*, SC26-4921
- *DFSMS/MVS Using Magnetic Tapes*, SC26-4923
- *DFSMS/MVS Utilities*, SC26-4926
- *Service Level Reporter User's Guide: Reporting*, SH19-6530
- *DFSMS/MVS Object Access Method Application Programmer's Reference*, SC26-4917
- *DFSMS/MVS Object Access Method Planning, Installation, and Storage Administration Guide for Object Support*, SC26-4918
- *DFSORT Installation and Customization*, SC26-7041
- *DFSORT Getting Started with DFSORT R14*, SC26-4109
- *DFSMS/MVS Network File System Customization and Operation*, SC26-7029

- *DFSMS Optimizer User′s Guide and Reference*, SC26-7047
- *DFSMS/MVS DFSMSdss Storage Administration Guide*, SC26-4930
- *DFSMShsm Storage Administration Guide*, SH21-1076
- *DFSMShsm Storage Administration Reference*, SH21-1075
- *DFSMS/MVS Network File System User′s Guide*, SC26-7028
- *DFSMS/MVS DFSMSrmm Guide and Reference*, SC26-4931
- *DFSMS/MVS DFSMSrmm Implementation and Customization Guide*, SC26-4932
- *MVS/ESA Storage Management Library Managing Data*, SC26-3124
- *MVS/ESA Storage Management Library Managing Storage Groups*, SC26-3125
- *MVS/ESA Storage Management Library Leading a Storage Administration Group*, SC26-3126.
- *DFSMS/MVS Using the Interactive Storage Management Facility*, SC26-4911
- *ADSTAR Distributed Storage Manager for MVS Administrator′s Guide*, GC35-0277
- *OS/390 MVS Programming: Assembler Services Guide*, GC28-1762
- *OS/390 MVS Programming: Resource Recovery*, GC28-1739
- *OS/390 MVS Setting Up a Sysplex*, GC28-1779
- *OS/390 MVS Sysplex Services Guide*, GC28-1771
- *OS/390 Parallel Sysplex Systems Management*, GC28-1861
- *OS/390 MVS Systems Codes*, GC28-1780
- *OS/390 MVS System Messages Volume 1*, GC28-1784
- *OS/390 MVS System Messages Volume 2*, GC28-1785
- *OS/390 MVS System Messages Volume 3*, GC28-1786
- *OS/390 MVS System Messages Volume 4*, GC28-1787
- *OS/390 MVS System Messages Volume 5*, GC28-1788
- *OS/390 MVS Installation Exits*, SC28-1753
- *OS/390 MVS Diagnosis Reference*, SY28-1084
- *CICS User′s Handbook*, SX33-1188
- *CICS Diagnosis Guide*, LX33-6093
- *MQSeries for MVS/ESA Messages and Codes*, GC33-0819

How to get IBM Redbooks

This section explains how both customers and IBM employees can find out about IBM Redbooks, redpieces, and CD-ROMs. A form for ordering books and CD-ROMs by fax or e-mail is also provided.

- **Redbooks Web Site** http://www.redbooks.ibm.com/

  Search for, view, download, or order hardcopy/CD-ROM Redbooks from the Redbooks Web site. Also read redpieces and download additional materials (code samples or diskette/CD-ROM images) from this Redbooks site.

  Redpieces are Redbooks in progress; not all Redbooks become redpieces and sometimes just a few chapters will be published this way. The intent is to get the information out much quicker than the formal publishing process allows.

- **E-mail Orders**

  Send orders by e-mail including information from the redbook fax order form to:

In United States:	e-mail address: usib6fpl@ibmmail.com
Outside North America:	Contact information is in the ″How to Order″ section at this site: http://www.elink.ibmlink.ibm.com/pbl/pbl/

- **Telephone Orders**

United States (toll free)	1-800-879-2755
Canada (toll free)	1-800-IBM-4YOU
Outside North America	Country coordinator phone number is in the ″How to Order″ section at this site: http://www.elink.ibmlink.ibm.com/pbl/pbl/

- **Fax Orders**

United States (toll free)	1-800-445-9269
Canada	1-403-267-4455
Outside North America	Fax phone number is in the ″How to Order″ section at this site: http://www.elink.ibmlink.ibm.com/pbl/pbl/

This information was current at the time of publication, but is continually subject to change. The latest information may be found at the Redbooks Web site.

IBM Intranet for Employees

IBM employees may register for information on workshops, residencies, and Redbooks by accessing the IBM Intranet Web site at http://w3.itso.ibm.com/ and clicking the ITSO Mailing List button. Look in the Materials repository for workshops, presentations, papers, and Web pages developed and written by the ITSO technical professionals; click the Additional Materials button. Employees may access MyNews at http://w3.ibm.com/ for redbook, residency, and workshop announcements.

IBM Redbooks fax order form

Please send me the following:

Title	Order Number	Quantity

First name Last name

Company

Address

City Postal code Country

Telephone number Telefax number VAT number

- Invoice to customer number

- Credit card number

Credit card expiration date Card issued to Signature

We accept American Express, Diners, Eurocard, Master Card, and Visa. Payment by credit card not available in all countries. Signature mandatory for credit card payment.

Glossary

A

abend. Termination of a task before its completion because of an error condition that cannot be resolved by recovery facilities while the task executing.

ACB. Access method control block.

access. A specific type of interaction between a subject and an object that results in the flow of information from one to the other.

access authority. An authority that relates to a request for a type of access to protected resources. In RACF, the access authorities are NONE, READ, UPDATE, ALTER, and EXECUTE.

access list. A list within a profile of all authorized users and their access authorities.

access method control block (ACB).. A control block that links an application program to VTAM.

ACDS. Active control data set.

ACF/VTAM. An IBM licensed program that controls communication and the flow of data in an SNA network. It provides single-domain, multiple-domain, and interconnected network capability. VTAM runs under MVS (OS/VS1 and OS/VS2), VSE, and VM/SP and supports direct control application programs and subsystems such as VSE/POWER.

ACIF. (1) AFP conversion and indexing facility. (2) A PSF utility program that converts a print file into AFP, MO:DCA-P, creates an index file for input data, and collects resources used by an AFP document into separate file.

action message retention facility (AMRF). A facility that, when active, retains all action messages except those specified by the installation in the MPFLSTxx member in effect.

action message sequence number. A decimal number assigned to action messages.

Advanced Function Presentation (AFP). A set of licensed programs, together with user applications, that use the all-points-addressable concept to print on presentation devices. AFP includes creating, formatting, archiving, retrieving, viewing, distributing, and printing information.

Advanced Program-to-Program Communications (APPC). A set of inter-program communication services that support cooperative transaction processing in a SNA network.

AFP. Advanced Function Presentation.

AFP Printer Driver for Windows. A component of Infoprint Server for OS/390 that runs on a Windows 95 or Windows NT workstation and creates output in AFP format, for printing on AFP printers.

AFP Viewer plug-in for Windows. A component of Infoprint Server for OS/390 that runs on a Windows 95 or Windows NT workstation and allows you to view files in AFP format.

AIX operating system. IBM′s implementation of the UNIX operating system. The RS/6000 system, among others, runs the AIX operating system.

allocate. To assign a resource for use in performing a specific task.

alphanumeric character. A letter or a number.

amode. Addressing mode. A program attribute that can be specified (or defaulted) for each CSECT, load module, and load module alias. AMODE states the addressing mode that is expected to be in effect when the program is entered.

AMRF. action message retention facility

AOR. Application-owning region

APPC. Advanced Program-to-Program Communications

APPN. Advanced Peer-to-Peer Networking.

ASCII (American Standard Code for Information Interchange). The standard code, using a coded character set consisting of 7-bit coded characters (8-bit including parity check), that is used for information interchange among data processing systems, data communication systems, and associated equipment. The ASCII set consists of control characters and graphic characters.

audit. To review and examine the activities of a data processing system mainly to test the adequacy and effectiveness of procedures for data security and data accuracy.

authority. The right to access objects, resources, or functions.

authorization checking. The action of determining whether a user is permitted access to a RACF-protected resource.

Authorized Program Analysis Report (APAR). A request for correction of problem caused by a defect in a current unaltered release of a program.

authorized program facility (APF). A facility that permits identification of programs authorized to use restricted functions.

automated operations. Automated procedures to replace or simplify actions of operators in both systems and network operations.

AVR. Automatic volume recognition.

B

banner page. A page printed before the data set is printed.

basic mode. A central processor mode that does not use logical partitioning. Contrast with logically partitioned (LPAR) mode.

batch message processing (BMP) program. An IMS batch processing program that has access to online databases and message queues. BMPs run online, but like programs in a batch environment, they are started with job control language (JCL).

batch-oriented BMP program. A BMP program that has access to online databases and message queues while performing batch-type processing. A batch-oriented BMP does not access the IMS message queues for input or output. It can access online databases, GSAM databases, and MVS files for both input and output.

BMP. Batch message processing (BMP) program.

broadcast. (1) Transmission of the same data to all destinations. (2) Simultaneous transmission of data to more than one destination.

binary data. (1) Any data not intended for direct human reading. Binary data may contain unprintable characters, outside the range of text characters. (2) A type of data consisting of numeric values stored in bit patterns of 0s and 1s. Binary data can cause a large number to be placed in a smaller space of storage.

BIND. In SNA, a request to activate a session between two logical units (LUs).

buffer. A portion of storage used to hold input or output data temporarily.

buffered device. A device where the data is written to a hardware buffer in the device before it is placed on the paper (for example, IBM 3820).

burst. To separate continuous-forms paper into single sheets.

C

cache structure. A coupling facility structure that enables high-performance sharing of cached data by multisystem applications in a sysplex. Applications can use a cache structure to implement several different types of caching systems, including a store-through or a store-in cache.

carriage control character. An optional character in an input data record that specifies a write, space, or skip operation.

carriage return (CR). (1) A keystroke generally indicating the end of a command line. (2) In text data, the action that indicates to continue printing at the left margin of the next line. (3) A character that will cause printing to start at the beginning of the same physical line in which the carriage return occurred.

CART. Command and response token.

case-sensitive. Pertaining to the ability to distinguish between uppercase and lowercase letters.

catalog. (1) A directory of files and libraries, with reference to their locations. (2) To enter information about a file or a library into a (3) The collection of all data set indexes that are used by the control program to locate a volume containing a specific data set.

CBPDO. Custom Built Product Delivery Offering.

CEC. Synonym for central processor complex (CPC).

central processor (CP). The part of the computer that contains the sequencing and processing facilities for instruction execution, initial program load, and other machine operations.

central processor complex (CPC). A physical collection of hardware that includes main storage, one or more central processors, timers, and channels.

CFRM. Coupling facility resource management.

channel-to-channel (CTC). Refers to the communication (transfer of data between programs on opposite sides of a channel-to-channel adapter (CTCA

channel-to-channel adapter (CTCA). An input/output device that is used a program in one system to communicate with a program in another system.

checkpoint. (1) A place in a routine where a check, or a recording of data for restart purposes, is performed. (2) A point at which information about the status of a job and the system can be recorded so that the job step can be restarted later.

checkpoint write. Any write to the checkpoint data set. A general term for the primary, intermediate, and final writes that update any checkpoint data set.

CICS. Customer Information Control System.

CICSplex. A group of connected CICS regions.

CICSPlex SM. CICSPlex System Manager

client. A functional unit that receives shared services from a server. See also client-server.

client-server. In TCP/IP, the model of interaction in distributed data processing in which a program at one site sends a request to a program at another site and awaits a response. The requesting program is called a client; the answering program is called a server.

CMOS. Complementary metal-oxide semiconductor.

CNGRPxx. The SYS1.PARMLIB member that defines console groups for the system or sysplex.

code page. (1) A table showing codes assigned to character sets. (2) An assignment of graphic characters and control function meanings to all code points. (3) Arrays of code points representing characters that establish ordinal sequence (numeric order) of characters. (4) A particular assignment of hexadecimal identifiers to graphic elements.

code point. A 1-byte code representing one of 256 potential characters.

coexistence. Two or more systems at different levels (for example, software, service or operational levels) that share resources. Coexistence includes the ability of a system to respond in the following ways to a new function that was introduced on another system with which it shares resources: ignore a new function, terminate gracefully, support a new function.

command and response token (CART). A parameter on WTO, WTOR, MGCRE, and certain TSO/E commands and REXX execs that allows you to link commands and their associated message responses.

command prefix facility (CPF). An MVS facility that allows you to define and control subsystem and other command prefixes for use in a sysplex.

COMMDS. Communications data set.

complementary metal-oxide semiconductor (CMOS). A technology that combines the electrical properties of positive and negative voltage requirements to use considerably less power than other types of semiconductors.

connection. In TCP/IP, the path between two protocol applications that provides reliable data stream delivery service. In Internet communications, a connection extends from a TCP application on one syste system to a TCP application on another system.

console. That part of a computer used for communication between the operator or user and the computer.

console group. In MVS, a group of consoles defined in CNGRPxx, each of whose members can serve as an alternate console in console or hardcopy recovery or as a console to display synchronous messages.

CONSOLxx. The SYS1.PARMLIB member used to define message handling, command processing, and MCS consoles.

control unit. Synonymous with device control unit.

conversation. A logical connection between two programs over an LU type 6.2 session that allows them to communicate with each other while processing a transaction.

conversational. Pertaining to a program or a system that carries on a dialog with a terminal user, alternately accepting input and then responding to the input quickly enough for the user to maintain a train of thought.

copy group. One or more copies of a page of paper. Each copy can have modifications, such as text suppression, page position, forms flash, and overlays.

couple data set. A data set that is created through the XCF couple data set format utility and, depending on its designated type, is shared by some or all of the MVS systems in a sysplex. See also sysplex couple data set.

coupling facility. A special logical partition that provides high-speed caching, list processing, and locking functions in a sysplex.

coupling facility channel.. A high bandwidth fiber optic channel that provides the high-speed connectivity required for data sharing between a coupling facility and the central processor complexes directly attached to it.

coupling services. In a sysplex, the functions of XCF that transfer data and status between members of a group residing on one or more MVS systems in the sysplex.

CP. Central processor.

CPC. Central processor complex.

CPF. Command prefix facility.

cross-system coupling facility (XCF). XCF is a component of MVS that provides functions to support cooperation between authorized programs running within a sysplex.

cryptography. The transformation of data to conceal its meaning.

cryptographic key. A parameter that determines cryptographic transformations between plaintext and ciphertext.

CTC. Channel-to-channel.

Customer Information Control System (CICS). An IBM licensed program tha that enables transactions entered at remote terminals to be processed concurrently by user-written application programs. It includes facilities for building, using, and maintaining databases.

D

DAE. Dump analysis and elimination.

daemon. A program that runs unattended to perform a standard service.

DASD. Direct access storage device.

data definition name. The name of a data definition (DD) statement, which corresponds to a data control block that contains the same name. Abbreviated as ddname.

data definition (DD) statement. A job control statement that describes a data set associated with a particular job step.

data integrity. The condition that exists as long as accidental or intentional destruction, alteration, or loss of data does not occur.

data set. The major unit of data storage and retrieval, consisting of a collection of data in one of several prescribed arrangements and described by control information to which the system has access.

data set label. (1) A collection of information that describes the attributes of a data set and is normally stored on the same volume as the data set. (2) A general term for data set control blocks and tape data set labels.

data set separator pages. Those pages of printed output that delimit data sets.

data sharing. The ability of concurrent subsystems (such as DB2 or IMS DB) or application programs to directly access and change the same data while maintaining data integrity.

data stream. (1) All information (data and control commands) sent over a data link usually in a single read or write operation. (2) A continuous stream of data elements being transmitted, or intended for transmission, in character or binary-digit form, using a defined format.

DBCS. Double-byte character set.

DBCTL. IMS Database Control.

DBRC. Database Recovery Control.

DB2. DATABASE 2 for MVS/ESA.

DB2 data sharing group. A collection of one or more concurrent DB2 subsystems that directly access and change the same data while maintaining data integrity.

DB2 PM. DB2 Performance Monitor.

deallocate. To release a resource that is assigned to a specific task.

default. A value, attribute, or option that is assumed when no alternative is specified by the user.

destination node. The node that provides application services to an authorized external user.

device control unit. A hardware device that controls the reading, writing, or displaying of data at one or more input/output devices or terminals.

device number. The unique number assigned to an external device.

device type. The general name for a kind of device; for example, 3330.

DFSMS. Data Facility Storage Management Subsystem.

direct access storage device (DASD). A device in which the access time effectively independent of the location of the data.

directory. (1) A type of file containing the names and controlling information for other files or other directories. Directories can also contain subdirectories, which can contain subdirectories of their own. (2) A file that contains directory entries. No two directory entries in the same directory can have the same name. (POSIX.1). (3) A file that points to files and to other directories. (4) An index used by a control program to locate blocks of data that are stored in separate areas of a data set in direct access storage.

display console. In MVS, an MCS console whose input/output function you can control.

DLL filter. A filter that provides one or more of these functions in a dynamic load library - init(), prolog(), process(), epilog(), and term(). See cfilter.h and cfilter.c in the /usr/lpp/Printsrv/samples/ directory for more information. See also filter. Contrast with DLL filter.

DOM. An MVS macro that removes outstanding WTORs or action messages that have been queued to a console end-of-tape-marker. A marker on a magnetic tape used to indicate the end of the permissible recording area, for example, a photo-reflective strip a transparent section of tape, or a particular bit pattern.

dotted decimal notation. The syntactical representation for a 32-bit integer that consists of four 8-bit numbers written in base 10 with periods (dots) separating them. It is used to represent IP addresses.

double-byte character set (DBCS). A set of characters in which each character is represented by a two-bytes code. Languages such as Japanese, Chinese, and Korean, which contain more symbols than can be represented by 256 code points, require double-byte character sets. Because each character requires two bytes, the typing, display, and printing of DBCS characters requires hardware and programs that support DBCS. Contrast with single-byte character set.

drain. Allowing a printer to complete its current work before stopping the device.

E

entry area. In MVS, the part of a console screen where operators can enter commands or command responses.

EMIF. ESCON Multiple Image Facility.

Enterprise Systems Connection (ESCON). A set of products and services that provides a dynamically connected environment using optical cables as a transmission medium.

EPDM. IBM SystemView Enterprise Performance Data Manager/MVS.

ESCD. ESCON Director.

ESCM. ESCON Manager. The licensed program System Automation for OS/390 includes all of the function previosuly provided by ESCM.

ESCON. Enterprise Systems Connection.

ETR. External Time Reference. See also Sysplex Timer.

extended MCS console. In MVS, a console other than an MCS console from which operators or programs can issue MVS commands and receive messages. An extended MCS console is defined through an OPERPARM segment.

F

FMID. Function modification identifier. The IBM release-specific product identifier such as HJE6610 for OS/390 Release 1 JES2.

FOR. File-owning region.

frame. For a System/390 microprocessor cluster, a frame contains one or two central processor complexes (CPCs), support elements, and AC power distribution.

FSS. functional subsystem. An address space uniquely identified as performing a specific function related to the JES. An example of an FSS is the program Print Services Facility that operates the 3800 Model 3 an 38xx printers.

functional subsystem (FSS). An address space uniquely identified as performing a specific function related to the JES.

functional subsystem application (FSA). The functional application program managed by the functional subsystem.

functional subsystem interface (FSI). The interface through which JES2 JES3 communicate with the functional subsystem.

G

gateway node. A node that is an interface between networks.

generalized trace facility (GTF). Like system trace, gathers information used to determine and diagnose problems that occur during system operation. Unlike system trace, however, GTF can be tailored to record very specific system and user program events.

global access checking. The ability to allow an installation to establish an in-storage table of default values for authorization levels for selected resources.

global resource serialization. A function that provides an MVS serialization mechanism for resources (typically data sets) across multiple MVS images.

global resource serialization complex. One or more MVS systems that use global resource serialization to serialize access to shared resources (such as data sets on shared DASD volumes).

group. A collection of RACF users who can share access authorities for protected resources.

GTF. Generalized trace facility.

H

hardcopy log. In systems with multiple console support or a graphic console, a permanent record of system activity.

hardware. Physical equipment, as opposed to the computer program or method of use; for example, mechanical, magnetic, electrical, or electronic devices. Contrast with software.

hardware configuration dialog. In MVS, a panel program that is part of the hardware configuration definition. The program allows an installation to define devices for MVS system configurations.

Hardware Management Console. A console used to monitor and control hardware such as the System/390 microprocessors.

HCD. Hardware Configuration Definition.

highly parallel. Refers to multiple systems operating in parallel, each of which can have multiple processors. See also n-way.

I

ICMF. Integrated Coupling Migration Facility.

IMS. Information Management System.

IMS DB. Information Management System Database Manager.

IMS DB data sharing group. A collection of one or more concurrent IMS DB subsystems that directly access and change the same data while maintaining data integrity.

IMS TM. Information Management System Transaction Manager.

initial program load (IPL). The initialization procedure that causes an operating system to begin operation.

instruction line. In MVS, the part of the console screen that contains messages about console control and input errors.

internal reader. A facility that transfers jobs to the job entry subsystem (JES2 or JES3).

IOCDS. Input/output configuration data set.

IOCP. Input/output configuration program.

IODF. Input/output definition file.

IPL. Initial program load.

IRLM. Internal resource lock manager.

ISPF. Interactive System Productivity Facility.

J

JES common coupling services. A set of macro-driven services that provide the communication interface between JES members of a sysplex. Synonymous with JES XCF.

JESXCF. JES cross-system coupling services. The MVS component, common to both JES2 and JES3, that provides the cross-system coupling services to either JES2 multi-access spool members or JES3 complex members, respectively.

JES2. An MVS subsystem that receives jobs into the system, converts them to internal format, selects them for execution, processes their output, and purges them from the system. In an installation with more than one processor, each JES2 processor independently controls its job input, scheduling, and output processing.

JES2 multi-access spool configuration. A multiple MVS system environment that consists of two or more JES2 processors sharing the same job queue and spool

JES3. An MVS subsystem that receives jobs into the system, converts them to internal format, selects them for execution, processes their output, and purges them from the system. In complexes that have several loosely-coupled processing units, the JES3 program manages processors so that the global processor exercises centralized control over the local processors and distributes jobs to them via a common job queue.

JES3 complex. A multiple MVS system environment that allows JES3 subsystem consoles and MCS consoles with a logical association to JES3 to receive messages and send commands across systems.

job entry subsystem (JES). A system facility for spooling, job queuing, and managing the scheduler work area.

job separator page data area (JSPA). A data area that contains job-level information for a data set. This information is used to generate job header, job trailer or data set header pages. The JSPA can be used by an installation-defined JES2 exit routine to duplicate the information currently in the JES2 separator page exit routine.

job separator pages. Those pages of printed output that delimit jobs.

K

keyword. A part of a command operand or SYS1.PARMLIB statement that consists of a specific character string (such as NAME= on the CONSOLE statement of CONSOLxx).

L

LIC. Licensed Internal Code.

list structure. A coupling facility structure that enables multisystem applications in a sysplex to share information organized as a set of lists or queues. A list structure consists of a set of lists and an optional lock table, which can be used for serializing resources in the list structure. Each list consists of a queue of list entries.

lock structure. A coupling facility structure that enables applications in a sysplex to implement customized locking protocols for serialization of application-defined resources. The lock structure supports shared, exclusive, and application-defined lock states, as well as generalized contention management and recovery protocols.

logical partition (LP). A subset of the processor hardware that is defined to support an operating system. See also logically partitioned (LPAR) mode.

logically partitioned (LPAR) mode. A central processor complex (CPC) power-on reset mode that enables use of the PR/SM feature and allows an operator to allocate CPC hardware resources (including central processors, central storage, expanded storage, and channel paths) among logical partitions. Contrast with basic mode.

logical unit (LU). In SNA, a port through which an end user accesses th SNA network in order to communicate with another end user and through which the end user accesses the functions provided by system services control points (SSCPs).

logical unit type 6.2. The SNA logical unit type that supports general communication between programs in a cooperative processing environment.

loosely coupled. A multisystem structure that requires a low degree of interaction and cooperation between multiple MVS images to process a workload. See also tightly coupled.

LP. Logical partition.

LPAR. Logically partitioned (mode).

M

MAS. Multi-access spool.

master console. In an MVS system or sysplex, the main console used for communication between the operator and the system from which all MVS commands can be entered. The first active console with AUTH(MASTER) defined becomes the master console in a system or sysplex.

master console authority. In a system or sysplex, a console defined with AUTH(MASTER) other than the master console from which all MVS commands can be entered.

master trace. A centralized data tracing facility of the master scheduler, used in servicing the message processing portions of MVS.

MCS. Multiple console support.

MCS console. A non-SNA device defined to MVS that is locally attached to an MVS system and is used to enter commands and receive messages.

member. A specific function (one or more modules/routines) of a multisystem application that is defined to XCF and assigned to a group by the multisystem application. A member resides on one system in the sysplex and can use XCF services to communicate (send and receive data) with other members of the same group.

message processing facility (MPF). A facility used to control message retention, suppression, and presentation.

message queue. A queue of messages that are waiting to be processed or waiting to be sent to a terminal.

message text. The part of a message consisting of the actual information that is routed to a user at a terminal or to a program.

microprocessor. A processor implemented on one or a small number of chips.

mixed complex. A global resource serialization complex in which one or more of the systems in the global resource serialization complex are not part of a multisystem sysplex.

MP. Multiprocessor.

MPF. Message processing facility.

MPFLSTxx. The SYS1.PARMLIB member that controls the message processing facility for the system.

MRO. Multiregion operation.

multiple console support (MCS). The operator interface in an MVS system.

multi-access spool (MAS). A complex of multiple processors running MVS/JES2 that share a common JES2 spool and JES2 checkpoint data set.

multiprocessing. The simultaneous execution of two or more computer programs or sequences of instructions. See also parallel processing.

multiprocessor (MP). A CPC that can be physically partitioned to form two operating processor complexes.

multisystem application. An application program that has various functions distributed across MVS images in a multisystem environment.

multisystem console support. Multiple console support for more than one system in a sysplex. Multisystem console support allows consoles on different systems in the sysplex to communicate with each other (send messages and receive commands)

multisystem environment. An environment in which two or more MVS images reside in one or more processors, and programs on one image can communicate with programs on the other images.

multisystem sysplex. A sysplex in which two or more MVS images are allowed to be initialized as part of the sysplex.

MVS image. A single occurrence of the MVS/ESA operating system that has the ability to process work.

MVS router. The MVS router is a system service that provides an installation with centralized control over system security processing.

MVS system. An MVS image together with its associated hardware, which collectively are often referred to simply as a system, or MVS system.

MVS/ESA. Multiple Virtual Storage/ESA.

MVSCP. MVS configuration program.

N

n-way. The number (n) of CPs in a CPC. For example, a 6-way CPC contains six CPs.

NIP. Nucleus initialization program.

NJE. Network job entry.

no-consoles condition. A condition in which the system is unable to access any full-capability console device.

nonstandard labels. Labels that do not conform to American National Standard or IBM System/370 standard label conventions.

nucleus initialization program (NIP). The stage of MVS that initializes the control program; it allows the operator to request last minute changes to certain options specified during initialization.

O

offline. Pertaining to equipment or devices not under control of the processor.

OLTP. Online transaction processing.

online. Pertaining to equipment or devices under control of the processor.

OPC/ESA. Operations Planning and Control.

operating system (OS). Software that controls the execution of programs and that may provide services such as resource allocation, scheduling, input/output control, and data management. Although operating systems are predominantly software, partial hardware implementations are possible.

operations log. In MVS, the operations log is a central record of communications and system problems for each system in a sysplex.

OPERLOG. The operations log.

OPERPARM. In MVS, a segment that contains information about console attributes for extended MCS consoles running on TSO/E.

OS/390. OS/390 is a network computing-ready, integrated operating system consisting of more than 50 base elements and integrated optional features delivered as a configured, tested system.

OS/390 Network File System. A base element of OS/390, that allows remote access to MVS host processor data from workstations, personal computers, or any other system on a TCP/IP network that is using client software for the Network File System protocol.

OS/390 UNIX System Services (OS/390 UNIX). The set of functions provided by the SHELL and UTILITIES, kernel, debugger, file system, C/C++ Run-Time Library, Language Environment, and other elements of the OS/390 operating system that allow users to write and run application programs that conform to UNIX standards.

P

parallel processing. The simultaneous processing of units of work by many servers. The units of work can be either transactions or subdivisions of large units of work (batch). See also highly parallel.

Parallel Sysplex. A sysplex that uses one or more coupling facilities.

partitionable CPC. A CPC that can be divided into 2 independent CPCs. See also physical partition, single-image mode, MP, side.

partitioned data set (PDS). A data set on direct access storage that is divided into partitions, called members, each of which can contain a program, part of a program, or data.

partitioned data set extended (PDSE). A system-managed data set that contains an indexed directory and members that are similar to the directory and members of partitioned data sets. A PDSE can be used instead of a partitioned data set.

password. A unique string of characters known to a computer system and to a user, who must specify the character string to gain access to a system and to the information stored within it.

permanent data set. A user-named data set that is normally retained for longer than the duration of a job or interactive session. Contrast with temporary data set.

PFK. Program function key.

PFK capability. On a display console, indicates that program function keys are supported and were specified at system generation.

PFKTABxx. The SYS1.PARMLIB member that controls the PFK table settings for MCS consoles in a system.

physical partition. Part of a CPC that operates as a CPC in its own right, with its own copy of the operating system.

physically partitioned (PP) configuration. A system configuration that allows the processor controller to use both central processor complex (CPC) sides as individual CPCs. The A-side of the processor controller controls side 0; the B-side of the processor controller controls side 1. Contrast with single-image (SI) configuration.

PR/SM. Processor Resource/Systems Manager.

Print Services Facility (PSF). The access method that supports the 3800 Printing Subsystem Models 3 and 8. PSF can interface either directly to a user′s application program or indirectly through the Job Entry Subsystem (JES) of MVS.

printer. (1) A device that writes output data from a system on paper or other media.

processor controller. Hardware that provides support and diagnostic functions for the central processors.

Processor Resource/Systems Manager (PR/SM). The feature that allows the processor to use several MVS images simultaneously and provides logical partitioning capability. See also LPAR.

profile. Data that describes the significant characteristics of a user, a group of users, or one or more computer resources.

program function key (PFK). A key on the keyboard of a display device that passes a signal to a program to call for a particular program operation.

program status word (PSW). A doubleword in main storage used to control the order in which instructions are executed, and to hold and indicate the status of the computing system in relation to a particular program.

pseudo-master console. A subsystem-allocatable console that has system command authority like that of an MCS master console.

PSW. Program status word.

R

RACF. See Resource Access Control Facility.

RAID. See redundant array of independent disk.

RAMAC Virtual Array (RVA) system. An online, random access disk array storage system composed of disk storage and control unit combined into a single frame.

read access. Permission to read information.

recording format. For a tape volume, the format of the data on the tape, for example, 18, 36, 128, or 256 tracks.

recovery. The process of rebuilding data after it has been damaged or destroyed, often by using a backup copy of the data or by reapplying transactions recorded in a log.

redundant array of independent disk (RAID). A disk subsystem architecture that combines two or more physical disk storage devices into a single logical device to achieve data redundancy.

remote operations. Operation of remote sites from a host system.

Resource Access Control Facility (RACF). An IBM-licensed program or a base element of OS/390, that provides for access control by identifying and verifying the users to the system, authorizing access to protected resources, logging the detected unauthorized attempts to enter the system and logging the detected accesses to protected resources.

restructured extended executor (REXX). A general-purpose, procedural language for end-user personal programming, designed for ease by both casual general users and computer professionals. It is also useful for application macros. REXX includes the capability of issuing commands to the underlying operating system from these macros and procedures. Features include powerful character-string manipulation, automatic data typing, manipulation of objects familiar to people, such as words, numbers, and names, and built-in interactive debugging.

REXX. See restructured extended executor.

RMF. Resource Measurement Facility.

rmode. Residency mode. A program attribute that can be specified (or defaulted) for each CSECT, load module, and load module alias. RMODE states the virtual storage location (either above 16 megabytes or anywhere in virtual storage) where the program should reside.

roll mode. The MCS console display mode that allows messages to roll of off the screen when a specified time interval elapses.

roll-deletable mode. The console display mode that allows messages to roll off the screen when a specified time interval elapses. Action messages remain at the top of the screen where operators can delete them.

routing. The assignment of the communications path by which a message will reach its destination.

routing code. A code assigned to an operator message and used to route the message to the proper console.

RVA. See RAMAC Virtual Array system.

S

SCDS. Source control data set.

SDSF. System Display and Search Facility.

shared DASD option. An option that enables independently operating computing systems to jointly use common data residing on shared direct access storage devices.

side. A part of a partitionable CPC that can run as a physical partition and is typically referred to as the A-side or the B-side.

single point of control. The characteristic a sysplex displays when you can accomplish a given set of tasks from a single workstation, even if you need multiple IBM and vendor products to accomplish that particular set of tasks.

single system image. The characteristic a product displays when multiple images of the product can be viewed and managed as one image.

single-image (SI) mode. A mode of operation for a multiprocessor (MP) system that allows it to function as one CPC. By definition, a uniprocessor (UP) operates in single-image mode. Contrast with physically partitioned (PP) configuration.

single-system sysplex. A sysplex in which only one MVS system is allowed to be initialized as part of the sysplex. In a single-system sysplex, XCF provides XCF services on the system but does not provide signalling services between MVS systems. See also multisystem sysplex, XCF-local mode.

SLR. Service Level Reporter.

small computer system interface (SCSI). A standard hardware interface that enables a variety of peripheral devices to communicate with one another.

SMF. System management facilities.

SMP/E. System Modification Program Extended.

SMS. Storage Management Subsystem.

SMS communication data set. The primary means of communication among systems governed by a single SMS configuration. The SMS communication data set (COMMDS) is a VSAM linear data set that contains the current utilization statistics for each system-managed volume, which SMS uses to help balance space usage among systems.

SMS configuration. The SMS definitions and routines that the Storage Management Subsystem uses to manage storage.

SMS system group. All systems in a sysplex that share the same SMS configuration and communications data sets, minus any systems in the sysplex that are defined individually in the SMS configuration.

software. (1) All or part of the programs, procedures, rules, and associated documentation of a data processing system. (2) Contrast with hardware. A set of programs, procedures, and, possibly, associated documentation concerned with the operation of a data processing system. For example, compilers, library routines, manuals, circuit diagrams. Contrast with hardware.

spanned record. A logical record contained in more than one block.

status-display console. An MCS console that can receive displays of system status but from which an operator cannot enter commands.

storage administrator. A person in the data processing center who is responsible for defining, implementing, and maintaining storage manageme policies.

storage class. A collection of storage attributes that identify performance goals and availability requirements, defined by the storage administrator, used to select a device that can meet those goals and requirements.

storage group. A collection of storage volumes and attributes, defined the storage administrator. The collections can be a group of DASD volume or tape volumes, or a group of DASD, optical, or tape volumes treated as single object storage hierarchy. See tape storage group.

storage management. The activities of data set allocation, placement, monitoring, migration, backup, recall, recovery, and deletion. These can be done either manually or by using automated processes. cThe Storage Management Subsystem automates these processes for you, while optimizing storage resources. See also Storage Management Subsystem.

Storage Management Subsystem (SMS). A DFSMS/MVS facility used to automate and centralize the management of storage. Using SMS, a storage administrator describes data allocation characteristics, performance and availability goals, backup and retention requirements, and storage requirements to the system through data class, storage class, management class, storage group, and ACS routine definitions.

storage subsystem. A storage control and its attached storage devices. See also tape subsystem.

structure. A construct used by MVS to map and manage storage on a coupling facility. See cache structure, list structure, and lock structure.

subsystem-allocatable console. A console managed by a subsystem like JES3 or NetView used to communicate with an MVS system.

subsystem interface (SSI). An MVS component that provides communication between MVS and JES.

supervisor call instruction (SVC). An instruction that interrupts a program being executed and passes control to the supervisor so that it can perform a specific service indicated by the instruction.

support element. A hardware unit that provides communications, monitoring, and diagnostic functions to a central processor complex (CPC).

SVC routine. A control program routine that performs or begins a contro program service specified by a supervisor call instruction.

symmetry. The characteristic of a sysplex where all systems, or certain subsets of the systems, have the same hardware and software configurations and share the same resources.

synchronous messages. WTO or WTOR messages issued by an MVS system during certain recovery situations.

SYSLOG. The system log data set.

sysplex. A set of MVS systems communicating and cooperating with each other through certain multisystem hardware components and software services to process customer workloads. See also MVS system, Parallel Sysplex.

sysplex couple data set. A couple data set that contains sysplex-wide data about systems, groups, and members that use XCF services. All MVS systems in a sysplex must have connectivity to the sysplex couple data set. See also couple data set.

Sysplex Timer. An IBM unit that synchronizes the time-of-day (TOD) clocks in multiple processors or processor sides. External Time Reference (ETR) is the MVS generic name for the IBM Sysplex Timer (9037).

system control element (SCE). Hardware that handles the transfer of data and control information associated with storage requests between the elements of the processor.

system console. In MVS, a console attached to the processor controller used to initialize an MVS system.

system log (SYSLOG). In MVS, the system log data set that includes all entries made by the WTL (write-to-log) macro as well as the hardcopy log. SYSLOG is maintained by JES in JES SPOOL space.

system management facilities (SMF). An optional control program feature of OS/390 and MVS that provides the means for gathering and recording information that can be used to evaluate system usage.

System Modification Program Extended (SMP/E). In addition to providing the services of SMP, SMP/E consolidates installation data, allows more flexibility in selecting changes to be installed, provides a dialog interface, and supports dynamic allocation of data sets.

Systems Network Architecture (SNA). A description of the logical structure, formats, protocols, and operational sequences for transmitting information units through, and controlling the configuration and operation of networks.

system trace. A chronological record of specific operating system events. The record is usually produced for debugging purposes.

T

temporary data set. A data set that is created and deleted in the same job.

terminal. A device, usually equipped with a keyboard and some kind of display, capable of sending and receiving information over a link.

terminal user. In systems with time-sharing, anyone who is eligible to log on.

tightly coupled. Multiple CPs that share storage and are controlled by a single copy of MVS. See also loosely coupled, tightly coupled multiprocessor.

tightly coupled multiprocessor. Any CPU with multiple CPs.

Time Sharing Option (TSO). An option on the operating system; for OS/390 the option provides interactive time sharing from remote terminals.

TOR. Terminal-owning region.

transaction. In APPC/MVS, a unit of work performed by one or more transaction programs, involving a specific set of input data and initiating a specific process or job.

transaction program (TP). For APPC/MVS, any program on MVS that issues APPC/MVS or CPI Communication calls, or is scheduled by the APPC/MVS transaction scheduler.

U

undelivered message. An action message or WTOR that cannot be queued for delivery to the expected console. MVS delivers these messages to any console with the UD console attribute in a system or sysplex.

uniprocessor (UP). A CPC that contains one CP and is not partitionable.

UP. Uniprocessor.

V

VM. Virtual Machine.

virtual telecommunications access method (VTAM). A set of programs that maintain control of the communication between terminals and application programs running under DOS/VS, OS/VS1, and OS/VS2 operating systems.

volume. (1) That portion of a single unit of storage which is accessible to a single read/write mechanism, for example, a drum, a disk pack, or part of a disk storage module. (2) A recording medium that is mounted and demounted as a unit, for example, a reel of magnetic tape, a disk pack, a data cell.

volume serial number. A number in a volume label that is assigned when a volume is prepared for use in the system.

volume table of contents (VTOC). A table on a direct access volume that describes each data set on the volume.

VSAM. Virtual Storage Access Method.

VTAM. Virtual Telecommunications Access Method.

VTOC. Volume table of contents.

W

wait state. Synonymous with waiting time.

waiting time. (1) The condition of a task that depends on one or more events in order to enter the ready condition. (2) The condition of a processing unit when all operations are suspended.

WLM. MVS workload management.

wrap mode. The console display mode that allows a separator line between old and new messages to move down a full screen as new messages are added. When the screen is filled and a new message is added, the separator line overlays the oldest message and the newest message appears immediately before the line.

write-to-log (WTL) message. A message sent to SYSLOG or the hardcopy log.

write-to-operator (WTO) message. A message sent to an operator console informing the operator of errors and system conditions that may need correcting.

write-to-operator-with-reply (WTOR) message. A message sent to an operator console informing the operator of errors and system conditions that may need correcting. The operator must enter a response.

WTL message. Write-to-log message

WTO message. Write-to-operator message

WTOR message. Write-to-operator-with-reply message.

X

XCF. Cross-system coupling facility.

XCF PR/SM policy. In a multisystem sysplex on PR/SM, the actions that XCF takes when one MVS system in the sysplex fails. This policy provides high availability for multisystem applications in the sysplex.

XCF-local mode. The state of a system in which XCF provides limited services on one system and does not provide signalling services between MVS systems. See also single-system sysplex.

XRF. Extended recovery facility.

IBM Redbooks evaluation

ABCs of OS/390 System Programming Volume 5
SG24-5655-00

Your feedback is very important to help us maintain the quality of IBM Redbooks. **Please complete this questionnaire and return it using one of the following methods:**

- Use the online evaluation form found at http://www.redbooks.ibm.com/
- Fax this form to: USA International Access Code + 1 914 432 8264
- Send your comments in an Internet note to redbook@us.ibm.com

Which of the following best describes you?
__Customer   __Business Partner   __Solution Developer   __IBM employee
__None of the above

Please rate your overall satisfaction with this book using the scale:
(1 = very good, 2 = good, 3 = average, 4 = poor, 5 = very poor)

Overall Satisfaction ____________

Please answer the following questions:

Was this redbook published in time for your needs? Yes____ No____

If no, please explain:
____

____

____

____

What other redbooks would you like to see published?
____

____

____

Comments/Suggestions: **(THANK YOU FOR YOUR FEEDBACK!)**
____

____

____

____

____

SG24-5655-00
Printed in the U.S.A.

ABCs of OS/390 System Programming Volume 5

SG24-5655-00