IBM

Merging Systems into a Sysplex

- Position for PSLC and WLC benefits
- Exploit System-level Resource Sharing
- Maximize the benefits of sysplex

Frank Kyne
Jeff Belot
Grant Bigham
Alberto Camara Jr.
Michael Ferguson
Gavin Foster
Roger Lowe
Mirian Minomizaki Sato
Graeme Simpson
Valeria Sokal
Feroni Suhood

Redbooks

ibm.com/redbooks

International Technical Support Organization

Merging Systems into a Sysplex

December 2002

SG24-6818-00

Take Note! Before using this information and the product it supports, be sure to read the general information in “Notices” on page xi.

First Edition (December 2002)

This edition applies to Version 1 Release 3 of z/OS.

Comments may be addressed to:
IBM Corporation, International Technical Support Organization
Dept. HYJ Mail Station P099
2455 South Road
Poughkeepsie, NY 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

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Trademarks

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

@server®	DFSORT™	OS/390®
@server®	Enterprise Storage Server®	Parallel Sysplex®
Redbooks (logo) ™	ESCON®	PR/SM™
ibm.com®	FICON®	Redbooks™
z/Architecture™	Hiperbatch™	Requisite®
z/OS®	IBM®	RACF®
zSeries®	IMS™	RAMAC®
Advanced Peer-to-Peer Networking®	IMS/ESA®	RMF™
CICS®	Language Environment®	S/390®
CICSPlex®	MQSeries®	Sysplex Timer®
DB2®	MVS™	SOM®
DFSMSdfp™	MVS/ESA™	SOMobjects®
DFSMSdss™	MVS/SP™	Tivoli®
DFSMShsm™	NetView®	VTAM®
DFSMSrmm™	OS/2®	WebSphere®

The following terms are trademarks of other companies:

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

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

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

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

Preface

This IBM® Redbook provides information to help Systems Programmers plan for merging systems into a sysplex. zSeries® systems are highly flexibile systems capable of processing many workloads. As a result, there are many things to consider when merging independent systems into the more closely integrated environment of a sysplex. This book will help you identify these issues in advance and thereby ensure a successful project.

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.

Frank Kyne is a Senior I/T Specialist at the International Technical Support Organization, Poughkeepsie Center. He has been an author of a number of other Parallel Sysplex® Redbooks™. Before joining the ITSO four years ago, Frank worked in IBM Global Services in Ireland as an MVS™ Systems Programmer.

Jeff Belot is an OS/390® Technical Consultant at IBM Global Services Australia. He has 28 years experience in the mainframe operating systems field. His areas of expertise include sysplex, performance and security.

Grant Bigham is a Senior OS/390 Technical Specialist in Australia. He has 10 years of experience in mainframe systems programming, the last 4 of which have been in IBM Global Services. More recently, Grant has worked as a Technical Consultant within IBM GSA. His areas of expertise include z/OS®, OS/390, and Enterprise Linux®.

Alberto Camara Jr. is a Customer Support Representative working for Computer Associates in Brazil. He has 15 years of experience in operating systems and mainframe field. His areas of expertise include storage, performance and security.

Michael Ferguson is a Senior I/T specialist in the IBM Support centre in Australia. He has 16 years of experience in the OS/390 software field. His areas of expertise include Parallel Sysplex, OS/390 and Tivoli® OPC. He has been an author of a number of Parallel Sysplex Redbooks. Michael teaches both Parallel Syplex and Tivoli OPC courses in the Asia Pacific region, as well as providing consulting services to customers in these areas.

Gavin Foster is an OS/390 Technical Consultant working for IBM Global Services in Australia. He has 16 years of experience in the mainframe operating systems field. His areas of expertise include systems programming and consultancy on system design, upgrade strategies and platform deployment.

Roger Lowe is a Senior S/390® Technical Consultant in the Professional Services division of Independent Systems Integrators, an IBM Large Systems Business Partner in Australia. He has 18 years of experience in the operating systems and mainframe field. His areas of expertise include the implementation and configuration of the OS/390 and z/OS operating system and Parallel Sysplex.

Mirian Minomizaki Sato is a Customer Support Representative working for Computer Associates in Brazil. She has 12 years of experience in operating systems and mainframe field. She holds a degree in Data Processing from Faculdade de Tecnologia de Sao Paulo. Her area of expertise is life cycle management.

Graeme Simpson is a Senior OS390 Technical Specialist in Australia. He has 23 years of experience in mainframe systems programming. He has worked at IBM Global Services for 7 years. His areas of expertise include OS/390 and IBM Storage Products.

Valeria Sokal is a System Programmer in a large IBM customer in Brazil. She has 12 years of experience working on many aspects of MVS, including building and managing very large sysplexes. She has been involved in writing a number of other IBM Redbooks.

Feroni Suhood is a Senior Performance Analyst at IBM Global Services Australia. He has 20 years experience in the mainframe operating systems field. His areas of expertise include sysplex, performance and hardware evaluation.

Thanks to the following people for their contributions to this project:

Rich Conway
International Technical Support Organization, Poughkeepsie Center

Robert Haimowitz
International Technical Support Organization, Poughkeepsie Center

Jay Aiken
IBM USA

Cy Atkinson
IBM USA

Paola Bari
IBM USA

Frank Becker
Sparkassen-Informatik-Services West GmbH, Germany

Charlie Burger
IBM USA

Jose Castano
IBM USA

Angelo Corridori
IBM USA

Vic Cross
Independent Systems Integrators, Australia

Greg Daynes
IBM USA

Jerry Dearing
IBM USA

Dave Draper
IBM UK

Scott Fagen
IBM USA

Kingsley Furneaux
IBM UK

Sue Hamner
IBM USA

Johnathan Harter
IBM USA

Evan Haruta
IBM USA

Axel Hirschfeld
IBM Germany

Gayle Huntling
IBM USA

Michael P Kasper
IBM USA

John Kinn
IBM USA

Paul M. Koniar
Metavante Corporation, USA

Matti Laakso
IBM Finland

Tony Langan
IBM Canada

Jim McCoy
IBM USA

Jeff Miller
IBM USA

Bruce Minton
CSC USA

Marcy Nechemias
IBM USA

Mark Noonan
IBM Australia

Bill Richardson
IBM USA

Alvaro Salla
Maffei Informática, Brazil

Sim Schindel
IBM Turkey

Norbert Schlumberger
IBM Germany

William Schoen
IBM USA

Gregory Silliman
IBM USA

Nat Stephenson III
IBM USA

Dave Sudlik
IBM USA

Kenneth Trowell
IBM Australia

Susan Van Berkel
IBM Canada

Geert Van de Putte
IBM Belgium

Tom Wasik
IBM USA

Bob Watling
IBM UK

Gail Whistance
IBM USA

Mike Wood
IBM UK

Bob Wright
IBM USA

Dave Yackel
IBM USA

Comments welcome

Your comments are important to us!

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

- Use the online **Contact us** review redbook form found at:

  `ibm.com/redbooks`

- Send your comments in an Internet note to:

  `redbook@us.ibm.com`

- Mail your comments to the address on page ii.

1

Introduction

This chapter discusses the reason for producing this book and introduces the structure used in most of the subsequent chapters. It also describes some terms used throughout the book, and some assumptions that were used in its production. Summary information is provided about which system components must be merged as a set, and the sequence of those merges.

1.1 Why this book was produced

As customer implementation of the sysplex concept has become widespread, together with the acceptance that S/390, and subsequently zSeries, provides an outstanding general purpose computing platform, both for legacy and e-business applications, there has been a growing trend to consolidate into smaller numbers of single system images. In this context, a "single system image" may consist of a single OS/390 or z/OS system, or a number of systems in a sysplex.

While no two consolidation exercises are identical, there are a number of aspects that apply to many, if not all, situations. This book was produced in an effort to make these consolidations easier to plan and implement, and to avoid customers having to constantly "reinvent the wheel".

This document discusses the things to be considered when an existing system (or group of systems) is to be moved into an existing sysplex. It does *not* cover the considerations for moving applications from one system image to another. Such moves are more common and generally less complex than moving a complete system into a sysplex.

1.2 Starting and ending points

There are many things to consider when moving a system into a sysplex. While it is not possible to cover every consideration for every consolidation project, we have attempted to address issues that are likely to arise in most projects. In this document, we discuss product merges, such as when merging two RACF® databases, as well as more ephemeral aspects, such as data set naming conventions.

There are endless ways to configure z/OS environments, and it is unlikely that any two customer's environments are exactly the same. However, in order to write this book, we had to make some assumptions about the starting configuration, and the objectives for doing the merge.

We have assumed that the current configuration is as follows:

- An existing sysplex, configured as a single system image. By this we mean that all the systems are sharing a single sysres, a single catalog structure, a single security database, and so on. Throughout this document we use the term "target sysplex" to refer to this set of systems.
- Another system that currently does not share anything with the target sysplex. This system may run on another CPC in the same installation and has its own sysres, catalogs, security database, and so on. We use the term "incoming system" to refer to this system.

  If the incoming system is part of a sysplex, we assume that all those systems are in a single system image, and therefore can be moved as if they were one system. If the systems are *not* in a single system image (that is, they each have their own security database and so on), then you will need to go through this exercise separately for each individual system.

Before you can proceed with the merge project, there is a very fundamental decision that needs to be made: are you going to share all your user data between all the systems? This decision needs to be based on many criteria, including:

- Who uses the system? For example, if the users of the incoming system work for a different company than the current users of the target sysplex systems, you will probably *not* want to share data between the different systems.

- Do you plan on implementing data sharing and workload balancing across all the systems in the target sysplex, either now or at some time in the future? If so, you would need to have all DASD volumes accessible to all systems.
- Are there duplicate high level qualifiers on the user volumes? The easiest way to check this is to do an IDCAMS LISTCAT ALIAS against the Master Catalogs of the incoming system and the target sysplex, and check for duplicate HLQs that are not referring to the same files. This should be obvious if the same alias in the two Master Catalogs points to two different user catalogs. Obviously you will have some duplicates for system files, but you must check for user data. If there are duplicates, this will make it more complex to share all volumes, especially if there are many duplicates. This is discussed in more detail in 10.4, “Sharing Master Catalogs” on page 150.
- If you are going to share all the volumes between all systems, realize that this means that you are probably also going to need a single security environment, single JESplex, single HSM, and single SMS. A single Master Catalog is desirable, but not a necessity.
- Having access to all DASD from all systems gives you the capability to potentially restart any work on any system. If there is a planned outage, the applications from the affected system can be moved and run on one of the other systems. If you are not exploiting data sharing, this is an attractive way of minimizing the impact of planned (or unplanned) outages. If you want this capability, all DASD must be accessible from all systems.

An additional important consideration that we have to point out is that the original concept of a sysplex is *a group of systems with similar characteristics, similar service level objectives, and similar workloads, sharing data between the systems and doing dynamic workload balancing across all systems*. While this is the ideal Parallel Sysplex implementation which will allow a customer to derive maximum benefit fromt the technology,, we realize that many customers may not, for business or technical reasons, be able to implement it. Therefore, asymmetric configurations where some of the workloads on each system are not shared are certainly supported. However, if you find that you have a Parallel Sysplex in which the majority of systems and/or workloads are completely disjoint that—that is, they have nothing in common—then you should give additional consideration as to whether those systems should really reside in the same sysplex.

Specifically, sysplex was *not* designed to support subplexes—that is, subsets of the sysplex that have nothing in common with the other members of the sysplex except that they share the sysplex couple data sets and Sysplex Timer®. For example, the design point is not to have development and production systems in the same sysplex. While some products do support the idea of subplexes (DFSMShsm™, for example), others do not (TCP/IP, for example).

While Parallel Sysplex provides the capability to deliver higher application availability than any other solution, the closer relationship between the systems in a sysplex mean that it is possible for a problem on one system to have an impact on other members of the sysplex. Such problems, while rare, are more likely to arise if the sysplex consists of widely disparate systems (for example, test and production). For the highest levels of availability, therefore, we recommend against mixing very different types of systems, that are completely unrelated, in the same sysplex.

On the other hand, the pricing mechanisms implemented through Parallel Sysplex License Charging (PSLC) and Workload License Charging (WLC) do unfortunately provide a financial incentive to place as many systems as possible in the same sysplex.

At the end of the day, if you are in the position of deciding whether to merge a completely unrelated system into a sysplex, you need to balance the financial and technical (if any) benefits of the merge against the possible impact such a move could have on the availability of all the systems in the sysplex.

Similarly, if the merge would result in a very large number of systems in the sysplex (and those systems are *not* all related to each other), you need to consider the impact to your business if a sysplex outage were to take out all those systems. While the software savings could be significant (and are easily calculated), the cost of the loss of all systems could also be significant (although more difficult to calculate, unfortunately).

Once you have decided how the user DASD are going to be handled, and have satisfied yourself with the availability and financial aspects, you are in a position to start planning for the merge. This document is designed to help you through the process of merging each of the affected components. However, because there are so many potential end points (ranging from sharing nothing to sharing everything), we had to make some assumptions about the objective for doing the merge, and how things will be configured when the exercise is complete. We describe these assumptions in the following sections.

1.2.1 BronzePlex

Some customers will want to move systems that are completely unrelated into a sysplex simply to get the benefits of PSLC or WLC charging. In this case, there will be practically no sharing between the incoming system and the other systems in the target sysplex. This is a typical outsourcing configuration, where the sysplex consists of systems from different customers, and there is no sharing of anything (except the minimal sharing required to be part of a sysplex) between the systems. We have used the term “BronzePlex” to describe this type of sysplex.

1.2.2 GoldPlex

Other customers may wish to move the incoming system into the sysplex, and do some fairly basic sharing, such as sharing of sysres volumes, for example. In this case, the final configuration might consist of more than one JES2 MAS, and two logical DASD pools, each of which is only accessed by a subset of the systems in the sysplex. We have included in this configuration all of the components that can easily be merged, and do not require a number of other components to be merged at the same time. This configuration provides more benefits, in terms of improved system management, than the BronzePlex, so we have called this configuration a “GoldPlex”.

1.2.3 PlatinumPlex

The third configuration we have considered is where the objective is to maximize the benefits of sysplex. In this case, after the merge is complete, the incoming system will be sharing everything with all the other members of the sysplex. So there will be a single shared sysres between all the systems, a single JES MAS, a single security environment, a single automation focal point, and basically just one of everything in the sysplex. This configuration provides the maximum in systems management benefits and efficiency, so we have called it the “PlatinumPlex”.

Depending on which type of plex you want to achieve and which products you use, some or all of the chapters in this book will apply to you. Table 1-1 contains a list of the topics addressed in the chapters and indicates which ones apply to each plex type. However, we recommend that even if your objective is a BronzePlex you should still read all the chapters that address products in your configuration, to be sure you have not overlooked anything that could impact your particular environment.

Table 1-1 Product considerations by target environment type

Component	BronzePlex	GoldPlex	PlatinumPlex
Couple data sets and Coupling Facilities	X	X	X
System Logger	X	X	X
WLM	X	X	X
GRS	X	X	X
Language Environment®		X	X
SMF		X	X
JES2		X	X
Shared HFS			X
Parmlib/Proclib		X	X
VTAM®	X	X	X
TCP	X	X	X
RACF			X
SMS			X
HSM			X
RMM			X
OPC	X	X	X
Automated operations	X	X	X
Physical considerations	X	X	X
Data Set Naming conventions		X	X
Software Licensing	X	X	X
Operations	X	X	X
Maintenance	X	X	X
Exits and usermods	X	X	X

In addition, in each chapter there is a table of considerations specific to that topic. One of the columns in that table is headed “TYPE”, and the meaning of the symbols in that column are as follows:

B This consideration applies if the target environment is a BronzePlex.

G This consideration applies if the target environment is a GoldPlex.

P This consideration applies if the target environment is a PlatinumPlex.

Some considerations apply across all target environments, and some only apply to a subset of environments.

We also thought it would be helpful to identify up front a suggested sequence for merging the various components. For example, for some components, most of the merge work can and should be done in advance of the day the system is actually moved into the sysplex. The merge of other components must happen immediately prior to the system joining the sysplex.

And still others can be merged at some time after the system joins the sysplex. In Table 1-2 on page 6, we list each of the components addressed in this book, and indicate when the merge work can take place. This table assumes that your target environment is a PlatinumPlex. If your target is one of the other environments, refer to the specific chapter for more information.

Table 1-2 Timing of merge activities by component

Component	Sequence
Couple Data Sets and CFs	Before
System Logger	Before and immediately preceding
WLM	Before
GRS	Before
Language Environment	Before
SMF	Before and after
JES2	Before and immediately preceding
Shared HFS	After
Shared system data sets	Before
VTAM	Before
TCP	Before
RACF	Before and immediately preceding
SMS	Before and immediately preceding
HSM	Before and immediately preceding
RMM	Before and immediately preceding
OPC	Before and after
Automated operations	Before
Physical considerations	Before
Data Set Naming conventions	Before
Software Licensing	Before
Operations	Before and immediately preceding
Maintenance	Before
Exits and usermods	Before

In Table 1-2 on page 6, the entries in the “Sequence” column have the following meanings:

Before The bulk of the work required to merge these components can and should be carried out in the weeks and months preceding the day the incoming system is brought up as a member of the target sysplex.

Immediately preceding At least some of the work to merge this component must be done in the short period between when the incoming system is shut down, and when it is IPLed as a member of the target sysplex.

After The work required to complete the merge for this component can actually be postponed until after the incoming system is IPLed into the target sysplex.

Note: Regardless of whether your target environment is a BronzePlex, a GoldPlex, or a PlatinumPlex, the incoming system *must* be IPLed to join the sysplex for the first time. However, as long as you have done the required planning and preparation work, there should be no need to IPL any of the existing systems in the target sysplex in order to bring the incoming system into the sysplex.

1.3 Structure of each chapter

There are basically two types of chapters in this book. A number are product-related, discussing, for example, how you would move to a single RACFplex. To make this document more consistent and easier to use, we have tried to use a single format for those chapters.

The other chapters are not related to a specific product, and may address something more esoteric, like data set naming conventions, or operator procedures. It does not make sense to apply the same structure to those chapters as to the product ones, so each of those chapters will be structured in a way that is suitable for that particular topic.

The product chapters each consist of four sections as described below.

1.3.1 Are multiple subplexes supported or desirable

For some products, such as RACF, it is possible to have a sysplex with more than one subplex (called a RACFplex in the case of RACF, for example). In some cases, this may be a perfectly acceptable way to run on an ongoing basis. In some other cases, it simply may not be possible to have more than one subplex per sysplex—GRSplex is an example, where only one GRS complex can exist within a sysplex.

In this section in each of the relevant chapters, we discuss whether it is possible to run with multiple subplexes, and if so, the advantages and disadvantages of running one or multiple subplexes.

1.3.2 Considerations checklist

Assuming that you decide to run with just one subplex, this section in each chapter provides a checklist of things to consider. We felt that a checklist format is easier to use in helping to quickly identify any areas that may be a concern in your environment.

The checklist is followed by explanatory notes where required.

1.3.3 Implementation methodology

There are often a number of ways of moving to a single shared environment. Some of these may be safer to implement, or require fewer steps. This section provides a recommended methodology for merging disparate environments into a single shared environment. An example might be a tape management system—you might merge them by adding entries from one tape management system to the other, or you might effect the merge by offloading the databases, doing a merge of the flat files using special utilities, then reloading the resulting files into a new database. Both methods may be valid, but one may be a lower risk approach. In this section, we allow you to benefit from the experience of those that have been through this process before.

1.3.4 Tools and documentation

In this section, we describe tools that may be available to help you do the merge, and tell you where they may be obtained. In addition, if there is other documentation that would be useful during this exercise, we provide a list of those sources.

1.4 Terminology and assumptions

The majority of this book applies equally to OS/390 and z/OS. However, to make the book easier to read, we will simply say z/OS (as in, “IPL the z/OS systems”) when discussing something that applies to both z/OS and OS/390. If a particular point applies specifically to OS/390 or z/OS, we will state that clearly at the time.

This book discusses the considerations for merging existing systems into a sysplex. A *sysplex* is the set of systems that share a common sysplex CDS, and all have the same sysplex name. A *subplex* is a subset of those systems that have something in common. Throughout this document, you will see a lot of use of the term *plex*. The following section discusses the subplexes that we talk about, and the terms that we use to refer to them.

1.4.1 ‘Plexes

In this document we frequently talk about the set of systems that share a particular resource. In order to avoid repetitive use of long-winded terms like “all the systems in the sysplex that share a single Master Catalog”, we have coined terms like CATALOGplex to describe this set of systems. The following are some of these terms we use later in this document:

CATALOGplex The set of systems in a sysplex that share a set of *user* catalogs.

DASDplex The set of systems in a sysplex that all share the same DASD volumes.

ECSplex The set of systems that are using Enhanced Catalog Sharing (ECS) to improve performance for a set of shared catalogs. All the systems sharing a given catalog with ECS *must* be in the same GRSplex.

GRSplex The set of systems that are in a single GRS complex and serialize a set of shared resources using either a GRS Ring or using the GRS Star structure in the Coupling Facility (CF).

JESplex The set of systems, either JES2 or JES3, that share a single spool. In JES2 terms, this would be a single MAS. In JES3 terms, this would be a Global/Local complex.

HFSplex An HFSplex is a collection of z/OS systems that share the OMVS Couple Data Set (CDS). The OMVS CDS contains the sysplex-wide mount table and information about all participating systems, and all mounted file systems in the HFSplex.

HMCplex The set of Central Processing Complexes (CPCs—also sometimes referred to as a CEC or CPU) that can be controlled from a single HMC. It is possible to have just one sysplex in a HMCplex, or many sysplex per HMCplex, or more than one HMCplex per sysplex.

HSMplex The set of systems that share a set of DFSMShsm Journals and CDSs.

OAMplex The set of OAM instances that are all in the same XCF group. The scope of the OAMplex must be the same as the DB2® data sharing

group that contains the information about the OAM objects used by those instances.

OPCplex The set of systems whose batch work is managed from a single OPC Controller. The OPCplex may consist of systems in more than one sysplex, and may even include non-MVS systems.

RACFplex The set of systems in a sysplex that share a single logical RACF database.

RMFplex The set of systems in a sysplex that are running RMF™ and are using the RMF sysplex data server. All such RMF address spaces in a sysplex will connect to the same XCF group and therefore be in the same RMFplex.

RMMplex The set of systems in a sysplex that share a single RMM CDS.

SMSplex The set of systems in a sysplex that share a single SMS ACDS, SCDS, and COMMDS.

TCPplex The set of systems in a sysplex whose TCP/IP stacks are connected to each other.

VTAMplex The set of systems in a sysplex that have active VTAM subsystems—in other words, the set of systems in the VTAMplex is the same as the set of systems in the sysplex (assuming that VTAM is active on every system).

WLMplex The set of systems in a sysplex that share the WLM CDS.

Table 1-3 summarizes how many of each of these 'plexes are possible in a single sysplex. Note that there are often relationships between various plexes - for example, if you have a single SMSplex, you would normally also have a single RACFplex. These relationships are discussed in more detail in the corresponding chapters of this book.

Table 1-3 Number of subplexes per sysplex

'Plex	1 per sysplex	>1 per sysplex
CATALOGplex		X
DASDplex		X
ECSplex	X	
GRSplex	X	
JESplex		X
HFSplex (if using sysplex HFS sharing)	X	
HMCplex		X
HSMplex		X
OAMplex		X
OPCIpex		X
RACFplex		X
RMFplex	X	
RMMplex		X
SMSplex		X

'Plex	1 per sysplex	>1 per sysplex
TCPplex	X	
VTAMplex	X	
WLMplex	X	

There are also other plexes, which are not covered in this book, such as BatchPipesPlex, CICSplex, DB2plex, IMSplex, MQplex, TapePlex, VSAM/RLSplex, and so on. However, we felt that data sharing has been adequately covered in other books, so we did not include CICS®, DB2, IMS™, MQSeries®, or VSAM/RLS in this book.

1.5 "Gotchas"

While adding completely new systems to an existing sysplex is a relatively trivial task, adding an existing system (complete with all its workloads and customization) to an existing sysplex can be quite complex. And you do not have to be aiming for a PlatinumPlex for this to be the case. In some ways, trying to establish a BronzePlex can be even more complex, depending on how your systems are set up prior to the merge and how stringent the requirements are to keep them apart.

In this section, we review some situations that can make the merge to a BronzePlex or GoldPlex configuration difficult or maybe even impossible. We felt it was important to highlight these situations up front, so that if any of them apply to you, you can investigate these particular aspects in more detail before you make a final decision about how to proceed.

1.5.1 Duplicate data set names

In principle, all system programmers know that duplicate data set names are bad news. They can lead to mistakes, complexity, and potentially even security or integrity problems. However, if the systems containing the duplicate names are completely separated, at least the problem can be contained.

However, what happens if the systems are no longer completely separated? Let's look at a BronzePlex (apparently the simplest of the three configurations) situation. In a BronzePlex, the systems in the two subplexes cannot see each other's DASD, and there are completely separate catalog structures, RACF databases, production control systems, and so on. However, we need to look at what is *not* separate.

The first thing is GRS. In a multi-system environment, you would always treat data sets as global resources, meaning that any time you want to update a data set, GRS will check that no one else in the GRSplex (either on the same system or another system) is using that data set. However, what happens if you have a situation like that shown in Figure 1-1?

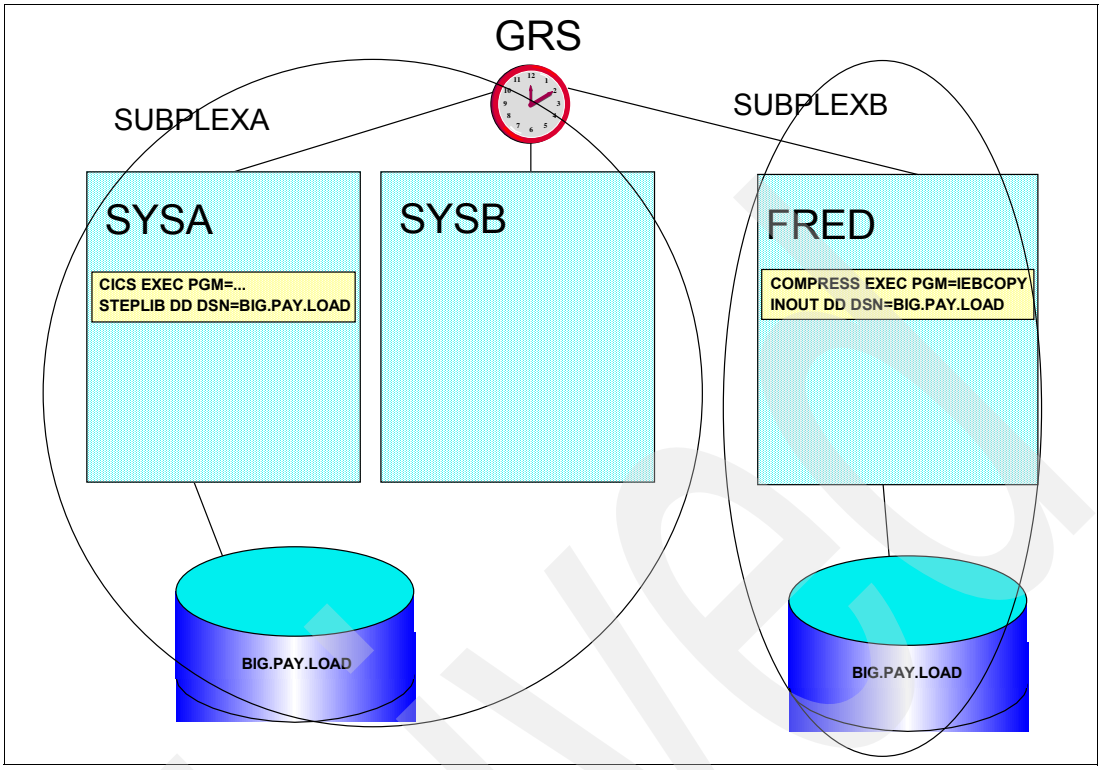


Figure 1-1 Multiple subplexes within a sysplex

In this case, systems SYSA and SYSB are in one subplex, SUBPLEXA (these might be the original target sysplex systems). System FRED is in another subplex, SUBPLEXB (this might be what was the incoming system). There are two data sets called BIG.PAY.LOAD, one that is used by system FRED, and another that is shared between systems SYSA and SYSB.

If the SYSA version of the data set is being updated by a job on SYSA, and someone on system FRED tries to update the FRED version of the data set, GRS will make the FRED update wait because it thinks someone on SYSA is already updating the same data set.

This is called *false contention*, and while it can potentially happen on any resource that is serialized by GRS, it is far more likely to happen if you have many duplicate data set names, or even if you have a small number of duplicate names, but those data sets are used very frequently.

To identify if this is likely to cause a problem in your installation, the first thing to do is to size the magnitude of the problem. A good place to start is by looking for duplicate aliases in the Master Catalogs. If you do not have a “clean” Master Catalog (that is, the Master Catalog contains many data sets other than those contained on the sysres volumes), then you should also check for duplicate data set entries in the Master Catalogs. If these checks indicate that duplicates may exist, further investigation is required. You might try running DCOLLECT against all the volumes in each subplex and all the DFSMShsm Migration Control Data Sets in each subplex, then use something like ICETOOL to display any duplicate records. While this is not ideal (it only reflects the situation at the instant the job is run, and it does not read information from the catalogs, such as for tape data sets), at least it will give you a starting place. For more information about the use of DCOLLECT, refer to *z/OS DFSMS Access Methods Services for Catalogs*, SC26-7394.

1.5.2 Duplicate volsers

Just as duplicate data set names are something you should avoid at all costs, a similar concern is duplicate volsers, whether they are on DASD or tape. The chapters on merging DFSMShsm and DFSMSrmm™ discuss this further; however, if your target environment is a BronzePlex or a GoldPlex, you may not be merging those components and therefore may not read that material. Duplicate volsers can cause confusion, operational errors, delayed IPLs (because of duplicate volser messages, each of which must be responded to manually), and potentially, integrity problems.

Another concern with duplicate volsers is in relation to PDSE extended sharing. Depending on how the library is being accessed, there will be normal ENQ-type serialization mechanisms—PDF, for example, serializes on the data set and member name. However, the PDSE code has its own serialization mechanism in addition to any used by the application.

Unlike PDF, however, the PDSE code does not use the data set name for serialization; rather, it uses the volser and the TTR of the DSCB to serialize access to the data set during create, delete, and member update-in-place processing. Therefore, it is possible that two different PDSE data sets, with different names, that reside on the same place on volumes with duplicate volsers could experience false contention during certain processes.

1.5.3 TCP/IP

There have been many enhancements to TCP/IP in an OS/390 and z/OS environment to improve both performance and availability. Many of these enhancements depend on a TCP/IP capability known as Dynamic XCF.

In addition to enabling these new enhancements, however, Dynamic XCF also automatically connects all TCP/IP stacks that indicate that they wish to use this feature. If you do not wish the TCP/IP stack on the incoming system to connect to the TCP/IP stacks on the target sysplex systems, then you cannot use Dynamic XCF on the incoming system. If the “incoming system” is actually just a single system, then this may not be a significant problem, however if the “incoming system” actually consists of more than one system, and you would like to use the TCP/IP sysplex enhancements between those systems, then this may be an issue.

If this is a potential concern to you, refer to Chapter 12, “TCP/IP considerations” on page 175 before proceeding.

1.5.4 System Logger

In a BronzePlex, you typically wish to share as little as possible. In general, the only DASD people plan on sharing in this environment are the volumes containing the CDSs.

However, there is another consideration you must plan for if you have *any* Logger structures (note that we said *structures*, not *logstreams*) that are connected to both subplexes. For most logstreams, you can specify any name you like for the logstream, so it should be possible to set up Logger structures so that all the connectors are in a single subplex. For example, structure SUBA_CICS_DFLOG could contain all the CICS DFHLOG logstreams from SUBPLEXA, and structure SUBB_CICS_DFHLOG could contain all the CICS DFHLOG logstreams from SUBPLEXB. In a BronzePlex and maybe in a GoldPlex, the only structures that would be connected to by members of both subplexes should be those containing logstreams that have a fixed name, and therefore you can only have one per sysplex—at the time of writing, the only such logstreams are SYSPLEX.OPERLOG and SYSPLEX.LOGREC.ALLRECS.

The reason for avoiding multiple subplexes being connected to a single Logger structure is that it is possible for any system that is connected to a Logger structure to do offload processing for a logstream in that structure. Therefore, the DASD volumes that will contain the offload data sets for those logstreams must also be shared by all systems in the sysplex. And because the offload data sets must be cataloged, the user catalog or catalogs that will contain the data set entries must also be shared.

An additional consideration is how do you manage the offload data sets. You can't use DFSMShsm to migrate them unless you have a single HSMplex, and you would normally only have a single HSMplex in a PlatinumPlex. The reason for this is that if DFSMShsm in SUBPLEXA migrates the data set, the catalog will be updated to change the volser of the migrated data set to MIGRAT. If a system in SUBPLEXB needs to access data in the migrated offload data set, it will see the catalog entry indicating that the data set has been migrated, call DFSMShsm on that system to recall it, and the recall will fail because the data set was migrated by a DFSMShsm in a different HSMplex. *If you wish to use DFSMShsm to manage the Logger offload data sets, all systems that are connected to a Logger structure must be in the same HSMplex.*

Furthermore, if you currently place your offload data sets on SMS-managed volumes (as many customers do), you will have to discontinue this practice if all the systems in the new enlarged sysplex will not be in the same SMSplex.

Therefore, if you are considering a BronzePlex or a GoldPlex configuration, you must give some thought to how your Logger structures will be used (especially for OPERLOG and LOGREC), and how the offloaded Logger data will be managed.

1.5.5 Sysplex HFS sharing

UNIX® Systems Services uses a fixed XCF group name to implement sysplex HFS sharing (introduced in OS/390 2.9). As a result, all the systems that enable this feature can immediately see all the HFSs mounted on any other system in the sysplex that also has this feature enabled. If your target environment is a PlatinumPlex, this is exactly what you want, because in this environment, you want full sharing. However, if your target environment is a BronzePlex or a GoldPlex, where you do not want to share user data between the different systems, then this could present a problem.

If the incoming system is a single system, and you are moving into a BronzePlex or a GoldPlex, you should not have a problem. In this case, there is no reason to enable sysplex HFS sharing on the incoming system, so it will only be able to see its own HFSs, and the systems in the target sysplex will only be able to see their HFSs, even if they have enabled sysplex HFS sharing.

However, if the incoming system is actually a multi-system sysplex and you want to share the HFSs in that sysplex using sysplex HFS sharing, then you could have a problem if the systems in the target sysplex also want to use sysplex HFS sharing to share their HFSs. The fact that the DASD containing the incoming system's HFSs are offline to the target sysplex systems does not provide any protection because all I/Os for a given HFS (that is being shared using sysplex HFS sharing) are issued from a single system. So conceivably, you could have the volume containing all your HFSs online to just one system, but every system in the sysplex could still read and write to those HFS if all systems have enabled sysplex HFS sharing.

Therefore, there can only be one HFSplex per sysplex, and systems whose HFS data sets should not be accessible from other systems must *not* enable sysplex HFS sharing.

1.5.6 Legal implications

Even in a BronzePlex environment, where the incoming system can only see its own DASD, and the original members of the target sysplex can only see their own DASD, the fact that all systems are in the same sysplex means that some information is available to all members of the systems. One example is syslog data. The current design of z/OS is such that all messages from every system get sent to every other system in the sysplex. Another example is RMF—because RMF uses a fixed XCF group name, every RMF in the sysplex will automatically communicate with every other RMF in the sysplex. This means that if you are logged on to SYSA and have access to RMF, you can potentially get performance information about any of the other members of the sysplex.

In most situations, this visibility of information is not a problem. However, if the two systems belong to different companies, possibly in the defense or finance industries, there may be legal restrictions on this level of access to information from the other company.

1.6 Updates to this document

It is our hope that this document will become the definitive place that people refer to should they need to merge a system into a sysplex, or just merge some of the components that we cover. If you find any additional considerations that we have not covered in this book, send an e-mail to redbook@us.ibm.com, providing the number of this manual (SG24-6818), and as much detail as possible about your finding. While we cannot guarantee it, we will attempt to include that information in any subsequent updates to this book.

2

Sysplex infrastructure considerations

This chapter discusses the following aspects of moving a system into an existing sysplex:

- Is it necessary to merge sysplex infrastructure (that is, XCF) environments and definitions, or is it possible to have more than one XCFplex in a single sysplex?
- A checklist of things to consider when merging a system into a sysplex.
- A methodology for doing the merge.
- A list of tools and documentation that can assist with this exercise.

2.1 One or more XCF subplexes

It is *not* possible to have more than one XCF environment in a sysplex. The definition of a sysplex is: the set of systems that all have the same sysplex name, have XCF communication between the systems, and share a common set of sysplex CDSs (and sysplex-related CDSs, such as ARM, CFRM, and so on). Therefore, when you move a system into a sysplex, it *must* have the same sysplex name as the other systems, and it *must* share the same sysplex CDSs—you cannot continue to use any other sysplex CDSs that the system might have been using previously.

2.2 CDS considerations

There are some characteristics and considerations that apply across all the CDSs, and we will address these here, before we move on to more specific issues relating to each of the sysplex components.

One of the parameters used when allocating *any* CDS (ARM, CFRM, LOGR, sysplex, OMVS, WLM) is the number of systems in the sysplex. If you are increasing the number of systems in the sysplex, you need to review all of the CDSs to check the current setting of the MAXSYSTEM parameter, which can be different for each CDS. To check the setting of the parameter, issue the **D XCF,C,TYPE=cdstype** command, as shown in Figure 2-1, for each type of CDS in use in the target sysplex.

```
D XCF,C,TYPE=SYSPLEX
IXC358I  22.55.49  DISPLAY XCF 534
SYSPLEX COUPLE DATA SETS
PRIMARY    DSN: SYS1.XCF.CDS01
           VOLSER: #@$#X1     DEVN: 37AC
           FORMAT TOD         MAXSYSTEM MAXGROUP(PEAK) MAXMEMBER(PEAK)
           05/15/2002 22:09:38        8     100   (52)      203   (12)
           ADDITIONAL INFORMATION:
            ALL TYPES OF COUPLE DATA SETS SUPPORTED
            GRS STAR MODE IS SUPPORTED
ALTERNATE  DSN: SYS1.XCF.CDS02
           VOLSER: #@$#X2     DEVN: 37AD
           FORMAT TOD         MAXSYSTEM MAXGROUP       MAXMEMBER
           05/15/2002 22:09:43        8      100             203
           ADDITIONAL INFORMATION:
            ALL TYPES OF COUPLE DATA SETS SUPPORTED
            GRS STAR MODE IS SUPPORTED
```

Figure 2-1 Displaying MAXSYSTEM value for sysplex CDS

2.2.1 MAXSYSTEM CDS parameter

Be aware that increasing the MAXSYSTEM value in the sysplex CDS can have an effect on the size of the XCF signalling structures and the VSAM/RLS lock structure. For the VSAM/RLS lock structure, the size of each lock entry depends on the MAXSYSTEM value in this CDS, *not* on the number of systems *actually* active in the sysplex. For the XCF structures, the number of lists in the structures increases as the value of MAXSYSTEM increases. The size of the GRS lock structure is related to the MAXSYSTEM value in the CFRM CDS. In order to determine the correct sizes for these structures, you should use the CFSizer tool, using the new MAXSYSTEM value as the number of systems in the sysplex. The CFSizer tool can be found at the following URL:

http://www.ibm.com/servers/eserver/zseries/cfsizer/

2.2.2 General CDS guidelines

The CDSs already exist in the target sysplex before the merge, but here are some points that you should keep in mind before you continue with the merge:

- CDSs are allocated and formatted using the CDS format utility (IXCL1DSU).
- The ARM, CFRM, LOGR, and SFM CDSs are customized using the IXCMIAPU utility.
- For every primary CDS there should be an alternate of the same size or larger (but not smaller).
- It is *highly* recommended to also have a spare CDS for every primary CDS.

  When an alternate CDS replaces a primary, the original primary data set is deallocated, and there is no longer an alternate CDS. Because you should always have an alternate CDS, you should have a spare CDS (one for each CDS you plan to use) that can be added in as the new alternate as soon as the old alternate becomes the primary.

  Note that the spare CDSs should be formatted with size parameters at least equal to the *larger* of the two that are being used for primary and alternate. That way, the spare will always be large enough so that it can be brought into use as an alternate, when needed.
- The CDS cannot exist prior to formatting.

  The format utility will not reformat an existing data set. This prevents the accidental re-formatting of an active CDS. If you wish to reformat an existing CDS, you must first delete the existing data set before trying to format one with that name.
- To set up a new CDS, you *must* use the IXCL1DSU utility—never try to create a new CDS by copying an existing one.
- CDSs cannot expand into multiple extents.
- A CDS cannot span volumes because XCF does not support multi-volume data sets.
- A CDS can be used by only one sysplex.
- A CDS can share a volume with other data sets. However, if you decide to allocate a CDS on a volume with other data sets:
  - Avoid any volume that has RESERVEs issued against it.
  - Avoid volumes with data sets that have high I/O rates.
- Review the following performance and availability considerations:

  The placement of CDSs can affect performance, as well as availability. For maximum performance and availability, the CDS volumes should only contain CDSs.

  - The primary sysplex CDS should be on a different volume than the primary CFRM CDS. During CF recover processing, these two data sets get extremely busy. Placing them on the same volume can impact the elapsed time for recovery from a CF failure.
  - All other primary CDSs can reside on one volume, and all other alternate CDSs can reside on another volume, as shown in Example 2-1 on page 18. However, be sure to monitor these data sets, and consider placing any high-activity data set on its own volume. (For example, the LOGR CDS might be a candidate for its own volume.).

Example 2-1 Placement of Couple Data Sets

```
Volume X Couple Data Sets     Volume Y Couple Data Sets     Volume Z Couple Data Sets
 ------------------------     -------------------------     ------------------------
    Sysplex (Primary)            Sysplex (Alternate)           Sysplex (Spare)
    CFRM (Alternate)             CFRM (Primary)                CFRM (Spare)
    SFM (Primary)                SFM (Alternate)               SFM (Spare)
    WLM (Primary)                WLM (Alternate)               WLM (Spare)
    ARM (Primary)                ARM (Alternate)               ARM (Spare)
    LOGR (Spare)                LOGR (Alternate)               LOGR(Primary)
...and so forth               ...and so forth               ...and so forth
```

- Place CDSs on volumes that are not subject to reserve/release contention or significant I/O contention from sources not related to the CDSs. This is true even if the I/O contention is sporadic.

  Note: it is vital that this recommendation is followed.

- Do not excessively over-specify parameter values when formatting CDSs, as this can result in larger-than-necessary CDSs. Large CDSs can elongate IPL times and cause performance problems during recovery processing. This recommendation covers the MAXSYSTEM, ITEM(GROUP) and ITEM(MEMBER) values.

- You can non-disruptively switch to a CDS that is the same size or larger at any time using the **SETXCF COUPLE,PSWITCH** command. However, if you wish to switch to a smaller CDS, a sysplex-wide IPL is required.

- Some sysplex monitoring tools may generate a large amount of I/O to the CFRM CDS, which may affect system performance. When using this type of tool, be aware of the performance implications.

- If a system cannot access a CDS for an extended period of time (for example, if the volume on which it resides is reserved by another system), z/OS switches to its alternate CDS. To minimize sysplex disruption, when you use DFDSS (or another data mover) to back up a volume with a CDS, it is recommended that you:

  - Convert the SYSVTOC reserves on the volume to global enqueues by creating an entry in the RESERVE conversion RNL for QNAME(SYSVTOC) RNAME(volser). For more information, see *z/OS MVS Planning: Global Resource Serialization,* SA22-7600.
  - Use logical volume backup processing so that the backups are done on a data set level, rather than on a track-image level. For more information, see *z/OS DFSMSdss Storage Administration Guide,* SC35-0423.

- Security considerations

  It is the responsibility of the installation to provide the security environment for the CDSs. Consider setting up a security profile specifically for the CDSs, and do not give any TSO users access to the profile. This protects against accidental deletion of CDSs.

2.3 Cross-System Coupling Facility

The Cross-System Coupling Facility (XCF) is a z/OS component that provides facilities for authorized programs to communicate with other programs on the same or different systems that are in the same sysplex. XCF is also used to control certain shared resources.

Authorized programs can join XCF groups and use XCF facilities to communicate with other group members. In addition to providing the communication facilities, XCF also informs members of the group when a member joins or leaves the group. XCF provides a relatively simple and high performance mechanism for programs to communicate with other programs resident in the same sysplex. As a result, there are many users of XCF, both IBM and non-IBM products.

A program on any system in the sysplex can communicate with another program on any other member of the sysplex as long as they are both connected to the same XCF group. Some products provide the ability to specify the name of the XCF group they will use, or provide the ability to control whether they connect to the group or not. Other products do not provide this level of control. This is an important consideration when determining whether your target sysplex will be a BronzePlex, a GoldPlex, or a PlatinumPlex—if you wish to maintain maximum separation between the incoming system and the other systems in the target sysplex, how products use XCF may have a bearing on that decision.

If your target environment is a BronzePlex, the incoming system must share the sysplex CDSs and take part in XCF signalling with the systems in the target sysplex. In fact, even if you share absolutely nothing else, to be a member of the sysplex, the incoming system *must* share the sysplex CDSs.

If your target environment is a GoldPlex, the incoming system will share the sysplex CDSs and take part in XCF signalling with the systems in the target sysplex.

Finally, if your target environment is a PlatinumPlex, the incoming system will again share the sysplex CDSs and take part in XCF signalling with the systems in the target sysplex.

2.3.1 Considerations for merging XCFs

This section contains, in checklist format, a list of items that must be considered when you decide to merge two or more XCFs.

Table 2-1 Considerations for merging XCF

Consideration	Note	Type	Done
Check MAXSYSTEM, GROUP, and MEMBER values	**1**	**B, G, P**	
Check COUPLExx member on incoming system	**2**	**B, G, P**	
Check XCF performance	**3**	**B, G, P**	
Check and update transport class definitions	**4**	**B, G, P**	
Understand who is using XCF groups	**5**	**B, G, P**	
Review XCF signalling path methods	**6**	**B, G, P**	

The “Type” specified in Table 2-1 relates to the sysplex target environment—**B** represents a BronzePlex, **G** represents a GoldPlex, and **P** represents a PlatinumPlex.

Notes indicated in Table 2-1 are described below:

1. If you need to increase the MAXSYSTEM value in the sysplex CDS, you will need to allocate a new CDS using the IXCL1DSU program. This is discussed in 2.2, “CDS considerations” on page 16.

In addition to the MAXSYSTEM value, there are other relevant attributes of the sysplex CDSs that need to be considered:

– ITEM NAME(GROUP) NUMBER( ) Specifies that the sysplex CDS supports group data. The NUMBER value includes not only the number of XCF groups needed for multisystem applications, but also the number of XCF groups that z/OS system components need. To determine the number of z/OS system groups currently in use, issue the **DISPLAY XCF,G** command on both the incoming system and the target sysplex. It is advisable to add a contingent number of groups for future growth. (Default=50, Minimum=10, Maximum=2045).

– ITEM NAME(MEMBER) NUMBER( ) Specifies that the sysplex CDS supports member data. The NUMBER value specifies the largest number of members allowed *in a single group*. Determine which group has the largest number of members and specify a value somewhat larger than this amount, allowing for the fact that bringing the incoming system into the target sysplex will probably increase the number of members in most, if not all, XCF groups. To verify the number of members in your groups at the moment, issue the **D XCF,G** command on both the incoming system and the target sysplex, as shown in Figure 2-2 on page 20. The number contained in brackets after the group name is the number of members in that group. In this example, the largest number of members in a group is 8, for the SYSMCS and SYSMCS2 groups. In a normal production environment, the groups with the largest numbers of members are usually those associated with CICS, JES3, and CONSOLE. (Minimum=8, Maximum=1023). To find out who is connected to a given group, use the **D XCF,GROUP,groupname,ALL** command.

```
D XCF,G
IXC331I  23.35.26  DISPLAY XCF 568
   GROUPS(SIZE):   ATRRRS(3)          COFVLFNO(3)        CSQGMQ$G(3)
                   CSQGPSMG(3)        D#$#(3)            DR$#IRLM(3)
                   DSNGSGA(3)         EZBTCPCS(3)        IDAVQUIO(3)
                   IGWXSGIS(5)        INGXSGA0(2)        IRRXCF00(3)
                   ISTCFS01(3)        ISTXCF(3)          ITSOIMS(1)
                   IXCLOOOC(3)        IXCLOOOD(3)        IXCLOOO1(3)
                   IXCLOOO4(3)        SYSBPX(3)          SYSDAE(7)
                   SYSENF(3)          SYSGRS(3)          SYSGRS2(1)
                   SYSIEFTS(3)        SYSIGW00(4)        SYSIGW01(4)
                   SYSIGW02(3)        SYSIGW03(3)        SYSIKJBC(3)
                   SYSIOS01(3)        SYSJES(3)          SYSMCS(8)
                   SYSMCS2(8)         SYSRMF(3)          SYSTTRC(3)
                   SYSWLM(3)          XCFJES2A(3)        XCFJES2K(1)
```

Figure 2-2 Displaying XCF group information

2. If your target environment is a BronzePlex, you will not be sharing the Master Catalog, so you must ensure that the CDSs are in the Master Catalog of each system. Remember that when the incoming system joins the target sysplex, it should use the same COUPLExx Parmlib member as the systems in the target sysplex, or at least use a member that is identical to the COUPLExx member used by those systems. If your target environment is a GoldPlex or a PlatinumPlex, you will be sharing SYS1.PARMLIB, so all systems should share the same COUPLExx member.

3. Prior to the merge, you should review your XCF performance in the target sysplex to ensure there are no hidden problems that could be exacerbated by adding another system to the sysplex. You should use WSC Flash 10011, *Parallel Sysplex Performance: XCF Performance Considerations,* to help identify the fields in RMF reports to check. If any problems are identified, they should be addressed prior to the merge.

After the merge, you should again use the XCF Activity Report to check the XCF performance in the target sysplex. Once again, if any problems are identified, they should be addressed now. In addition to the WSC Flash, you can use Chapter 6, “Tuning a Sysplex”, in *z/OS MVS Setting Up a Sysplex,* SA22-7625 to help with tuning XCF.

4. A transport class is z/OS's way of enabling you to associate XCF messages (based on similar signaling requirements) and then assign them signaling resources (signaling paths and message buffers). A transport class allows you to segregate message traffic according to the XCF group a message is related to, or according to the length of the message, or both. In general, we recommend using transport classes for one of the following reasons:

   – To assign signalling resources to groups of messages based on the message sizes; this ensures the most efficient use of the signalling resources.

   – To get information about the number of messages and the size of the messages associated with a given XCF group. We recommend that once you have obtained the required information, you remove the transport class and group the associated messages in with all other messages based solely on their size.

   For instance, you can define a transport class that optimizes signaling resources for XCF messages of a certain size. Messages larger than this size can still be processed, however, there may be an overhead in processing those messages. While it is possible to assign messages to a transport class based on their XCF group, we recommend using the message size rather than the XCF group as the mechanism to assign messages to a given transport class. The following are examples of how you would define two transport classes and assign all messages to one or the other based purely on message size (specifying GROUP(UNDESIG) indicates that messages from all XCF groups are candidates for selection for this message class):

   ```
   CLASSDEF CLASS(DEFSMALL) CLASSLEN(956) GROUP(UNDESIG)
   CLASSDEF CLASS(DEFAULT) CLASSLEN(16316) GROUP(UNDESIG)
   ```

   Transport classes are defined independently for each system in the sysplex using the CLASS keyword of the CLASSDEF statement or, after IPL, using the **SETXCF START,CLASSDEF** command.

Example 2-2 Syntax for CLASSDEF Statement of COUPLExx Parmlib Member

```
[CLASSDEF                                          ]
[          CLASS(class-name)                       ]
[         [CLASSLEN(class-length)          ]       ]
[         [GROUP(group-name[,group-name]...)]       ]
[         [MAXMSG(max-messages)            ]       ]
```

On a system, each transport class must have a unique name. The class name is used in system commands and shown in display output and reports.

By explicitly assigning an XCF group to a transport class, you give the group priority access to the signaling resources (signaling paths and message buffer space) of the transport class. All groups assigned to a transport class have equal access to the signaling resources of that class.

You should check to see if any transport classes are defined in either the incoming system or in the target sysplex. If there are, first check to see if they are still required—as a general rule, we recommend against having many transport classes unless absolutely necessary. If you determine that the transport classes are still necessary, you should then assess the impact of the incoming system, and make sure you add the appropriate definitions to the COUPLExx member that will be used by that system after it joins the sysplex.

5. Table 2-2 contains a list of XCF groups with the owner of each. The corresponding notes indicate whether the group name is fixed or not, and if not, where the group name is specified. If your target environment is a BronzePlex and you want to maintain strict segregation of the workloads, you should pay special attention to any exploiters that have fixed group names.

Table 2-2 XCF groups in the sysplex

Owner of the group	XCF group name	Note
APPC, ASCH	SYSATBxx	**A**
CICS MRO	DFHIR000	**B**
CICS VR	DWWCVRCM	**C**
CICSplex System Manager	Name not fixed	**D**
Console services	SYSMCS, SYSMCS2	**E**
DAE	SYSDAE	**F**
DB2	Name not fixed	**G**
DFSMS and PDSE sharing	SYSIGW00, SYSIGW01	**H**
ENF	SYSENF	**I**
Global Resource Serialization (GRS)	SYSGRS, SYSGRS2	**J**
HSM	ARCxxxxxx	**K**
IMS	Names not fixed	**L**
I/O Operations component of SA/390	ESCM	**M**
IOS	SYSIOSxx	**N**
IRLM	Name not fixed	**O**
JES2 Multi-Access Spool (MAS)	Name not fixed	**P**
JES3 complex	Name not fixed	**Q**
MQSeries	CSQGxxxx	**R**
Object Access Method (OAM)	xxxxxxxx	**S**
OMVS	SYSBPX	**T**
RACF	IRRXCF00	**U**
RMF	SYSRMF	**V**
RRS	ATRRRS	**W**
System Automation for OS/390 V2	INGXSGxx	**X**
Tape Sharing	SYSIEFTS	**Y**
TCP/IP	EZBTCPCS	**Z**
Tivoli Workload Scheduler (previously OPC)	Name not fixed	**aa**
Trace	SYSTTRC	**ab**
TSO Broadcast	SYSIKJBC	**ac**

Owner of the group	XCF group name	Note
VLF	COFVLFNO	**ad**
VSAM/RLS	IDAVQUI0, IGWXSGIS, SYSIGW01, SYSIGW02, SYSIGW03	**ae**
VTAM, TCP/IP	ISTXCF, ISTCFS01	**af**
WLM	SYSWLM	**ag**
XES	IXCLOxxx	**ah**

The notes in Table 2-2 on page 22 refer to the following points:

a. APPC has a unique group for each system in the sysplex, and only one system is in each group. The system generates the group name, which is in the format SYSATBxx. The APPC Component Control Block, ATBAPPCA, contains the group name for each system.

b. CICS will automatically connect to the DFHIR000 XCF group for MRO communications between systems. This mechanism is described in *CICS TS Installation Guide*, GC33-1681. The name of the XCF group used by CICS/MRO is fixed, so all the CICS regions in the sysplex using XCF for MRO communications will connect to the same group.

c. CICS VSAM Recovery connects to an XCF group with a fixed name of DWWCVRCM. It uses this group to gather information from other CICS VR address spaces in the sysplex in response to a **D SMS,CICSVR,ALL** or **D SMS,CICSVR,LOGSTREAMS** command.

d. CICSplex System Manager uses XCF to communicate between the CAS address spaces. The group name can be specified during customization of CICSplex SM, but the default is BBGROUP. For more information, refer to *CICSplex System Manager Setup and Administration-Volume 1*, SC33-0784.

e. MVS console services use two XCF groups—SYSMCS and SYSMCS2. SYSMCS is used to send WTOs, DOMs, and commands across systems, as well as to send data about types of consoles (MCS, SMCS, EMCS, and subsystem), and some miscellaneous data that CONSOLE needs on each system in the sysplex. SYSMCS2 is used for REPLYID processing, to keep track of which reply IDs are in use by each system in the sysplex.

f. The SYSDAE group is used by the Dump Analysis and Elimination component of MVS to pass information between systems about dumps that have been captured. This information is used to ensure that identical dumps are not captured multiple times across different systems in the sysplex.

g. DB2 uses its data sharing group name as the name of the XCF group that it joins. This name is defined during DB2 setup and must be unique within the sysplex.

h. DFSMS uses two XCF groups in support of sysplex sharing of PDSE data sets. The SYSIGW00 group is used for lock negotiation, and the SYSIGW01 group is used for status monitoring. The SMXC address space on each system attaches to these two groups.

i. The MVS Event Notification Facility (ENF) uses the SYSENF XCF group to send ENF signals from every system to every other system in the sysplex. The IEFSCHAS address space on every system automatically joins this group, and you have no control over the group name.

j. When GRS is operating in Ring mode, the SYSGRS group is used to communicate the RSA, acknowledgement signals, GRS join requests, and ECA requests. In this mode, there is no SYSGRS2 group.

   When GRS is operating in Star mode, the SYSGRS group is used to communicate GQSCAN requests and ECA requests. In this mode, the SYSGRS2 group only contains a list of the members of the GRSplex—it is not used for signalling.

k. HSM Secondary Host Promotion uses an XCF group to notify the members of the HSM XCF group should one of the members fail. The group name is ARCxxxxx, with the xxxxx value being specified via the HSM **SETSYS PLEXNAME** command. The default group name is ARCPLEX0. For more information, refer to *DFSMShsm Implementation and Customization Guide*, SC35-0418.

l. IMS OTMA uses XCF to communicate between IMS and the OTMA clients. The name of the XCF group used for this communication. The name is installation-dependent, and is specified on the GRNAME parameter in the IMS startup JCL.

   IMS also uses an XCF group to allow the Fast Database Recovery (FDBR) region to monitor the IMS regions it is responsible for. The group name is installation-dependent, and is specified on the GROUPNAME parameter in the IMS startup JCL. If not specified, if defaults to FDRimsid, where imsid is the IMSID of the IMS region that is being tracked.

   If using IMS Shared Message Queues, IMS also has an XCF group that is used by the IMS regions in the Shared Queues group. The name of the group is CSQxxxxx, where xxxxx is a one-to-five character value that is specified on the SQGROUP parameter in the DFSSQxxx Proclib member.

m. The I/O Operations component of System Automation for OS/390 (previously ESCON® Manager) attaches to a group with a fixed name of ESCM. All commands passed between I/O Ops on different systems are actually passed via VTAM (because I/O Ops supports systems spread over multiple sysplexes), however within a sysplex, I/O Ops uses the ESCM group to determine the VTAM name of the I/O Ops component on each system.

n. The I/O Supervisor component of MVS, via the IOSAS address space, connects to an XCF group with a name of SYSIOSxx. The xx suffix is determined dynamically when the system is IPLed. There is one SYSIOSxx group per LPAR cluster—so, if you had three LPAR clusters in the sysplex, there would be three SYSIOSxx groups. These groups are used by DCM to coordinate configuration changes across the systems in the LPAR cluster.

o. IRLM uses an XCF group to communicate between the IRLM subsystems in the data sharing group. The name of the IRLM XCF group is specified on the IRLMGRP parameter in the IRLM started task JCL. This can be any 8-character name you wish.

p. JES2 uses two XCF groups.

   One group is used for communicating between the members of the JES2 MAS. The default name for this group is the local node name defined on the NAME parameter of the local NODE(nnnn) initialization statement. We recommend that the default name be used, unless it conflicts with an existing XCF group name. Alternately, the XCF group name used by JES can be specified on the XCFGRPNM parameter of the MASDEF macro.

   The other group, with a fixed name of SYSJES, is used for collecting and sharing diagnostic information about JES/XCF interactions. It is also used in conjunction with TSO Generic Resource support.

q. JES3 also uses two XCF groups.

   One group is used for communicating between the members of the JES3 complex. The default name of this group is the node name defined by the NAME= keyword on the NJERMT initialization statement for the home node (HOME=YES), if one exists. If you have not defined any NJERMT statements, the default is N1. You can use the XCFGRPNM keyword of the OPTIONS initialization statement to override the default JES3 XCF group name.

   The other group, with a fixed name of SYSJES, is only used for collecting and sharing diagnostic information about JES/XCF interactions.

r. If you are using the MQSeries Shared Queues feature (available in MQ V5.2 and later), MQSeries joins an XCF group called CSQGxxxx, where xxxx is the MQ Shared Queues Group name. If there were two MQ Shared Queue Groups within the sysplex, there would be two XCF groups, and each MQ would only be attached to one of the groups.

s. DFSMS 1.5 introduced the concept of an OAMplex. By using this facility, a number of systems can access a shared set of objects, and provide the ability for one of the members of the OAMplex to take over ownership of the 3995 should the current owning system fail. The name of the OAMplex XCF group and the XCF member name of each OAM instance are both specified in the CBROAMxx member of Parmlib, and can be anything you wish.

   For more information about OAMplex support, refer to *z/OS DFSMS Object Access Method Planning, Installation, and Storage Administration Guide for Tape Libraries,* SC35-0427.

t. The UNIX System Services component uses an XCF group called SYSBPX to pass requests between systems that are sharing their HFSs using the Sysplex HFS Sharing capability introduced in OS/390 2.9. If the BPXPRMxx member used at IPL specifies SYSPLEX(YES), UNIX System Services will automatically join this group. If it specifies SYSPLEX(NO), it will not join the group, and that system will not be able to participate in the shared HFS group. The name of this group is fixed, so you can only have one HFS sharing group per sysplex. For more information on HFS sharing, refer to Chapter 8, “Shared HFS considerations” on page 115.

u. RACF optionally uses an XCF group to communicate between members of the RACF sysplex database sharing group. The name of this group is IRRXCF00 and is fixed, so there can only be one RACF sysplex database sharing group in the sysplex. Whether a given RACF instance joins the group or not is controlled by a setting in the ICFRDSNT module. This is discussed further in Chapter 13, “RACF considerations” on page 189.

v. RMF uses an XCF group called SYSRMF to pass performance information between all the RMF address spaces in the sysplex. This gives you the ability to be logged on to one system, but still see RMF information about any of the other systems in the sysplex. The name of the group is fixed, and RMF automatically joins the group. So, from any system in the sysplex, you could potentially see what is happening in any other system.

w. MVS Resource Recovery Services (RRS) uses an XCF group with a fixed name of ATRRRS to communicate between RRS instances on each system in the sysplex where RRS is started.

x. System Automation for OS/390 connects to an XCF group called INGXSGxx, where the xx suffix is defined using the GRPID parameter in the HSAPRMxx member of SYS1.PARMLIB. In addition, in System Automation for OS/390 V2, the Automation Manager address space connects to the same group. The suffix for the group must be specified in the INGXINIT member of DSIPARM for the Automation Agents. If there is

more than one System Automation for OS/390 instance in a single MVS system, each instance must have connect to a unique XCF group.

y. The automatic tape sharing feature of z/OS (in z/OS 1.2 and later) uses an XCF group with a fixed name of SYSIEFTS to pass information about tape drive allocations between all the members of the sysplex. You cannot control the name of the group, nor can you stop any system from connecting to the group.

z. All the TCP stacks in the sysplex join an XCF group called EZBTCPCS. This group is used by the stacks to exchange all the information they need to share for all the sysplex functions like Dynamic XCF, Dynamic VIPAs, Sysplex Distributor, and the like. The name of this group is fixed, so *all* the TCPs in the sysplex will potentially be able to communicate with each other using this group. This is discussed further in Chapter 12, “TCP/IP considerations” on page 175.

aa. The Tivoli Workload Scheduler for z/OS connects to one, or optionally, two XCF groups. One of the groups is used to communicate between the controller and trackers, and can be used to submit workload and control information to the trackers connected to the same group. The trackers use XCF services to transmit events back to the controller. The same group is used to give you the ability to define a hot-standby controller that takes over when XCF notifies it that the previous controller has failed.

In addition, if you are using the data store feature, you need a separate XCF group. The XCF Group names are specified on the XCFOPTS and DSTGROUP initialization statements. More information about these features can be found in *Tivoli Workload Scheduler for z/OS V8R1 Installation Guide*, SH19-4543.

ab. The MVS transaction trace facility uses an XCF group with a fixed name of SYSTTRC. Transaction trace provides a consolidated trace of key events for the execution path of application- or transaction-type work units running in a multi-system application environment. TRACE TT commands issued on any system in the sysplex affect all systems. The same filter sets are made active throughout the sysplex as a consequence of TRACE TT commands, and the systems using those filter sets are displayed in the output from the **DISPLAY TRACE,TT** command. The TRACE address space in every system in the sysplex automatically connects to the SYSTTRC XCF group, and the XCF group is used to communicate filter set information to all the systems in the sysplex. However, no trace data is sent over the XCF group, so it is unlikely that there are any security concerns due to the fact that all systems are connected to the same group.

ac. The SYSIKJBC XCF group, which has a fixed name, is used to send information to all the systems in the sysplex about whether the SYS1.BRODCAST data set is shared or not, and also to notify systems whenever there is an update to that data set. No actual data, such as the text of messages, for example, is passed through the group.

ad. The MVS Virtual Lookaside Facility (VLF) is used, together with LLA, to improve the performance of loading frequently used load modules and PDF members. During initialization, VLF on each system connects to an XCF group called COFVLFNO. This group is used by VLF to notify VLF on other systems when a member in a VLF-managed PDS is changed, thus ensuring that the other systems do not use in-storage, but downlevel versions of the member.

ae. VSAM/RLS uses a number of XCF groups, all of which have names that are determined by the system and cannot be controlled by the user. The IDAVQUIO group is used for RLS Quiesce Functions. The SYSIGW02 group is used for RLS Lock Manager Functions. The SYSIGW03 and IGWXSGIS groups are used for RLS Sharing Control Functions, and the SYSIGW01 group is for a common lock manager service which is used by both RLS and PDSE. These XCF groups are used to send messages around the sysplex in order to allow all of the SMSVSAM address spaces to communicate with each other. The groups will be connected to by any system where

VSAM/RLS is started. Note that you can only have one VSAM/RLS data sharing group per sysplex, because the lock structure and the XCF group names are all fixed.

af. VTAM joins an XCF group called ISTXCF to communicate with other VTAMs in the sysplex. There is no way to change the name of the group that VTAM joins, so if you have two independent groups of VTAMs in a single sysplex (for example, a production network and a test network), they will automatically communicate. The ISTXCF group is also used for TCP/IP session traffic between stacks in the same sysplex.

   VTAM also joins a second XCF group called ISTCFS01. ISTCFS01 is used to determine when another VTAM's address space terminates so the VTAM CF structures can be cleaned up. This structure is also used by MultiNode Persistent Session support to send messages for planned takeovers. Even if XCFINIT=NO is specified, VTAM will still join the ISTCFS01 group.

   More information about the interaction between VTAM and XCF is provided in Chapter 11, “VTAM considerations” on page 167.

ag. Every WLM in the sysplex joins an XCF group called SYSWLM. This group is used by all the WLMs to exchange information about service classes, service class periods, goal attainment, and so on. The name is fixed, and every WLM automatically connects to it, even when WLM is running in compatibility mode. This means that every WLM in the sysplex has information about what is happening on every other system in the sysplex. This is discussed further in Chapter 4, “WLM considerations” on page 51.

ah. XES creates one XCF group for each serialized list or lock structure that it connects to. The name of each group is IXCLOxxx, where xxx is a printable hex number. There is one member per structure connection.

6. It is possible to use both CF structures and CTCs for XCF signalling. If using CTCs, you must keep in mind that the formula for the number of the CTC definitions is n x (n-1), where n is the number of systems in the sysplex. For example, a sysplex composed of 12 systems has to define 132 CTCs (12 x 11).

   Another issue that should be considered is performance. Generally speaking, the performance of CF structures for XCF signalling is equivalent to that of CTCs, especially for XCF message rates below 1000 messages per second. If most XCF messages are large, CF structures may provide a performance benefit over CTCs. On the other hand, if most XCF messages are small, CTCs may provide better performance.

   Given that the performance for most environments is roughly equivalent, the use of CF structures for XCF signalling has one significant advantage—CF structures are *far* easier to manage than CTCs. To add another image to the sysplex, all you have to do is use the existing COUPLExx member on the new member, and possibly increase the size of the XCF CF structures. This will take a couple of minutes, compared to hours of work arranging the hardware cabling, updating HCD definitions, and setting up and updating COUPLExx members to add the new CTC definitions. If you decide to use CF structures instead of CTCs, just remember that you should have two CFs, and you should never place all your XCF structures in the same CF.

2.4 Coupling Facility Resource Management

The CFRM policy contains the definitions for the CFs used by the sysplex, and the structures within those CFs.

CFRM policies are stored in the CFRM CDS. In addition to the policies, the CDS also contains status information about the CFs.

The most complex scenario, from the point of view of merging the incoming system, is if the incoming system is attached to a CF and has some structures in that CF. In reality, it is unlikely that a standalone system would be attached to a CF, but we will cater for that situation in this section.

If your target environment is a BronzePlex, the incoming system's use of the CF is likely to be limited to sharing the XCF structures, IEFAUTOS for tape sharing (for systems prior to z/OS 1.2), and the GRS Star structure. You may also wish to use the OPERLOG and SYSTEM_LOGREC facilities, in which case you should refer to 1.5.4, "System Logger" on page 12. In addition, if the incoming system is using any CF structures prior to the merge, you will probably set up those in the CF, with only the incoming system using those structures.

If your target environment is a GoldPlex, the incoming system will probably share at least the XCF and GRS structures. It may also share other structures like the Enhanced Catalog Sharing structure, OPERLOG and LOGREC, JES2 checkpoint, and so on. Once again, if you are considering using OPERLOG and/or LOGREC, you should refer to 1.5.4, "System Logger" on page 12. In addition, if the incoming system is using any CF structures prior to the merge, you will probably set up those in the CF—in this case, it may or may not share those structures with the other systems in the target sysplex.

If your target environment is a PlatinumPlex, the incoming system will probably share all the structures in the CF with the other systems in the target sysplex. In addition, if the incoming system is using any CF structures prior to the merge, you will probably set up those in the CF—in this case, it may or may not share those structures with the other systems in the target sysplex.

2.4.1 Considerations for merging CFRM

This section contains, in checklist format, a list of items that must be considered when you decide to merge two or more sysplexes, at least one of which is using a CF prior to the merge.

Table 2-3 Considerations for merging CFRM

Consideration	Note	Type	Done
Ensure *all* systems in the sysplex have at least two links to all attached CFs.		**B, G, P**	
Check that the CFs have enough storage, and enough white space, to handle the additional structures as well as existing structures that will increase in size.		**B, G, P**	
Check link utilization if you are going to be sharing existing CF links.	**1**	**B, G, P**	
Check that performance of CFs are acceptable and that CFs have sufficient spare capacity to handle the increased load.	**2**	**B, G, P**	
Ensure target sysplex CFs have the appropriate CF Levels.	**3**	**B, G, P**	
Check that the CFRM CDS has an appropriate MAXSYSTEM value.	**4**	**B, G, P**	
Check the size of all structures that will be used by the incoming system.	**5**	**B, G, P**	
Ensure the target sysplex CFRM supports the same level of functionality as the CFRM in use in the incoming system.	**6**	**B, G, P**	

Consideration	Note	Type	Done
Move structure definitions from the incoming system CFRM policy to the target sysplex CFRM policy *as appropriate.*	**7**	**B, G, P**	
Check the options on any duplicate structures to ensure they are consistent.	**8**	**B, G, P**	
Consider the impact for structures that have fixed names.	**9**	**B, G, P**	
Do not merge Logger structures unless your target environment is a PlatinumPlex.	**10**	**B, G, P**	

The “Type” specified in Table 2-3 on page 28 relates to the sysplex target environment—**B** represents a BronzePlex, **G** represents a GoldPlex, and **P** represents a PlatinumPlex.

Notes indicated in Table 2-3 on page 28 are described below:

1. If the incoming system resides in an LPAR on a CPC that already contains other members of the target sysplex, it is likely that you will use MIF (Multiple Image Facility, formerly known as EMIF) to share the existing CF links between the incoming system and those other members. In this case, you should check the utilization of those links in advance of the merge. You use the PTH BUSY field in the RMF CF Subchannel Activity Report as an indicator of CF link utilization—the percentage of PTH BUSYs should not exceed 10%.

   After the merge, you should review the RMF reports again to check contention for CF paths. If the percentage of PTH BUSYs exceeds 10%, you should consider dedicating the CF links to each image or adding additional CF links.

2. Prior to the merge, check the performance of all the CFs in the target sysplex. We recommend that average utilization of CFs should not exceed 50%. This caters for the situation where you have to failover all your structures into a single CF, and you need to maintain acceptable performance in that situation. You want to ensure that not only is the current performance acceptable, but also that there is spare capacity sufficient to handle the estimated increase in utilization after the merge. For more information, refer to *OS/390 Parallel Sysplex Configuration Volume 2: Cookbook,* SG24-5638.

   After the merge, you should once again check the performance of all CFs to ensure the performance is still acceptable.

3. If the incoming system is using a CF prior to the merge, you must ensure that the CFs in the target sysplex have *at least* the same CF Level as those in use by the incoming system today. The CF Level dictates the functions available in the CF—for example, MQSeries shared queues require CF Level 9. More information about CF Levels, including which levels are supported on various CPC types is available at:

   `http://www.ibm.com/servers/eserver/zseries/pso/cftable.html`

4. If you need to increase the MAXSYSTEM value in the CFRM CDS, you will need to allocate a new CDS using the IXCL1DSU program. This is discussed in 2.2, “CDS considerations” on page 16.

5. In order to provide optimal performance and availability, it is important that all CF structures have appropriate sizes. Some structures are sensitive to the number of connected systems, or the MAXSYSTEM value as specified in the sysplex CDS. Therefore, you should review the structure sizes in any of the following situations:

   – A change in the number of connected systems
   – A change in the MAXSYSTEM value in the sysplex CDS
   – A significant change in workload

– A change in the CF Level—as additional functions are delivered, structure sizes tend to increase, so any time you change the CF Level you should review the structure sizes.

The only way to get an accurate size for a structure is to use the CFSizer tool available at:

http://www.ibm.com/servers/eserver/zseries/cfsizer/

Note: The PR/SM™ Planning Guide no longer can be used to calculate structure sizes. The formulas in that document have not been updated since CF Level 7, and will not give correct values for any CF Level higher than that. Also, the recommended structure values in the IBM Redbook *OS/390 Parallel Sysplex Configuration Volume 2: Cookbook,* SG24-5638 should also be ignored as they are based on old CF Levels.

6. If the incoming system is already using a CF before the merge, make sure the functions being used in that environment are also available in the target sysplex. The functions are usually specified in the CFRM CDS, and may have co-requisite CF Level requirements. You should list the options specified in the CFRM CDS in both incoming system and target sysplex. Compare the options that have been specified (System Managed Rebuild, System Managed Duplexing, and so on) and make sure the target sysplex will provide the required level of functionality. The sample job in Example 2-3 shows how the options are listed, and shows the available options.

Example 2-3 List of CFRM Couple Data Set

```
//ITSO01A JOB (ITSO01),'ITSO-LSKO51-R',
//         CLASS=A,NOTIFY=&SYSUID
//LOGDEFN  EXEC PGM=IXCMIAPU
//SYSPRINT DD SYSOUT=*
  DATA TYPE(CFRM) REPORT(YES)

  DATA TYPE(CFRM)
    ITEM NAME(POLICY) NUMBER(5)
    ITEM NAME(STR) NUMBER(200)
    ITEM NAME(CF) NUMBER(8)
    ITEM NAME(CONNECT) NUMBER(32)
    ITEM NAME(SMREBLD) NUMBER(1)
    ITEM NAME(SMDUPLEX) NUMBER(1)
...
```

7. If the incoming system is using structures that are not being used in the target sysplex, those structures (for example, the structure for a DB2 in the incoming system) must be added to the CFRM policy in the target sysplex.

   Other structures that may be in use in the incoming system, like the XCF structures, will not be moved over to the target sysplex because they will be replaced with shared ones that are already defined in the target sysplex. In that case, you must update any references to those structure names in the incoming system.

8. For structures that existed in both the incoming system and the target sysplex, like the XCF structures, make sure that the definition of those structures in the target sysplex is compatible with the definitions in the incoming system. For example, if the incoming system uses Auto Alter for a given structure and the target sysplex does not, you need to investigate and make a conscious decision about which option you will use.

9. Some structures have fixed names that you cannot control. Therefore, every system in the sysplex that wants to use that function must connect to the same structure instance. At the time of writing, the structures with fixed names are:

   – OPERLOG

   – LOGREC

   – GRS Lock structure, ISGLOCK

- VSAM/RLS Lock structure, IGWLOCK00
- RACF database sharing structures, IRRXCF00_B00n, IRRXCF00_P00n
- Tape sharing (prior to z/OS 1.2), IEFAUTOS
- Enhanced Catalog Sharing, SYSIGGCAS_ECS
- WLM Multi-System Enclaves
- WLM LPAR Cluster

If you use any of these structures in the incoming system, you need to assess the impact of moving the incoming system into the sysplex, especially in a BronzePlex environment. Additionally, for the Logger structures with fixed names (OPERLOG and LOGREC), if you are moving to a BronzePlex or a GoldPlex, you should review the discussion about shared DASD considerations in 1.5.4, “System Logger” on page 12.

10. If your target environment is a BronzePlex or a GoldPlex, you need to ensure that each Logger structure (except OPERLOG and LOGREC, as noted above) is only connected to by systems that are in the same subplex. Therefore, when merging your CFRM policies, pay particular heed *not* to merge Logger structures *except* if your target environment is a PlatinumPlex. This is discussed further in Chapter 3, “System Logger considerations” on page 37.

2.5 Sysplex Failure Management

Sysplex Failure Management (SFM) allows you to define a sysplex-wide policy that specifies the actions that z/OS is to take when certain failures occur in the sysplex. A number of situations might occur during the operation of a sysplex when one or more systems need to be removed so that the remaining sysplex members can continue to do work. The goal of SFM is to allow these reconfiguration decisions to be made and carried out with little or no operator involvement.

Some SFM actions are specified on the sysplex level, while others can be specified for individual systems. SFM controls actions for three scenarios:

1. If there is a loss of XCF signalling connectivity between a subset of the systems, SFM can decide, based on the weights you specify for the systems, which systems must be removed from the sysplex in order to restore full connectivity. This is controlled by the CONNFAIL keyword, and is enabled or disabled at the entire sysplex level.
2. In the event of the loss of a system, SFM can automatically partition that system out of the sysplex, releasing any resources it was holding, and allowing work on other systems to continue. This action can be specified at the individual system level. For example, you can tell SFM to automatically partition SYSA out of the sysplex if it fails, however to prompt the operator in case SYSB fails. This is controlled via the ISOLATETIME and PROMPT keywords. As a general rule however, we recommend that ISOLATETIME is always specified so that dead systems can be partitioned out of the sysplex as quickly as possible.
3. If you use PR/SM reconfiguration to move processor storage from one LPAR to another, in the event of a system failing, SFM will control that processing if you provide the appropriate definitions.

If your target environment is a BronzePlex, you need to decide how you want to use SFM. In a sysplex environment, it is vital that dead systems are partitioned out of the sysplex as quickly as possible. On the other hand, if a subset of the systems in the sysplex are running work that is completely independent of the other systems in the sysplex, you may prefer to have more control over what happens in the case of a failure. You may decide to specify CONNFAIL NO in this situation, which will cause the operator to be prompted to decide which system should be removed from the sysplex.

Figure 2-3 shows a typical configuration, where systems SYSA and SYSB were in the original target sysplex, and SYSC is the incoming system which is now a member of the sysplex. If the link between systems SYSB and SYSC fail, the sysplex can continue processing with SYSA and SYSB, or SYSA and SYSC. If the applications in SYSA and SYSB support data sharing and workload balancing, you may decide to remove SYSB as the applications from that system can continue to run on SYSA. On the other hand, if SYSA and SYSB contain production work, and SYSC only contains development work, you may decide to remove SYSC.

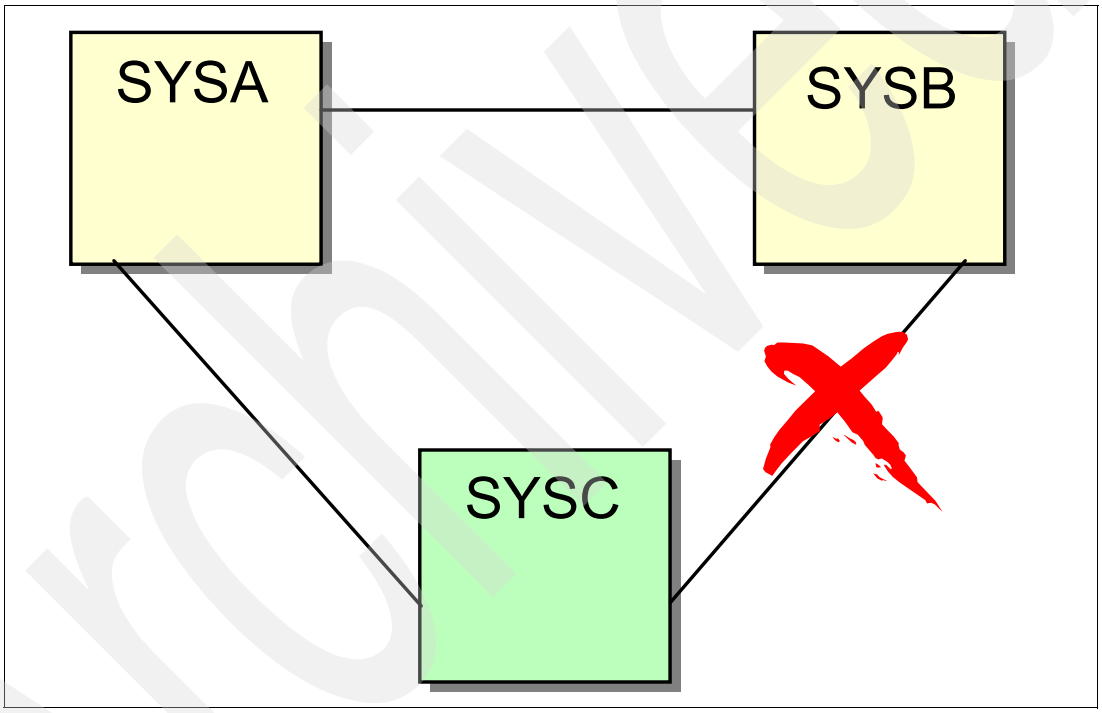


Figure 2-3 SFM CONNFAIL scenario

You can see that the decision about how to handle this scenario depends on your specific configuration and availability requirements. For each system, you then need to carefully evaluate the relative weight of that system within the sysplex, and whether you wish SFM to automatically partition it out of the sysplex (by specifying ISOLATETIME), or you wish to control that process manually (by specifying PROMPT). If you can assign a set of weights that accurately reflect the importance of the work in the sysplex, we recommend that you use ISOLATETIME.

If your target environment is a GoldPlex, your SFM decision will be based on the level or workload sharing across the systems in the sysplex. Once again, you want to partition dead systems out of the sysplex as quickly as possible. If the work is spread across all the systems in the sysplex, we recommend that you specify CONNFAIL YES, to quickly remove any

system that has lost XCF signalling connectivity, and allow the other systems to continue processing. For each system, you then need to carefully assign an appropriate value for WEIGHT and specify ISOLATETIME to automatically partition that system out of the sysplex in a timely manner.

Finally, if your target environment is a PlatinumPlex, which implies full workload balancing, your decision is actually easier. You should specify CONNFAIL YES, assign appropriate WEIGHT values to each LPAR, and specify ISOLATETIME.

2.5.1 Considerations for merging SFM

This section contains, in checklist format, a list of items that must be considered when you decide to merge two or more SFM policies, or add an additional system to an existing policy.

Table 2-4 Considerations for merging SFM

Consideration	Note	Type	Done
Update SFM policy to add name and weight of incoming system	1	B, G, P	
Check the setting of the CONNFAIL parameter	2	B, G, P	
Check use of PR/SM reconfiguration	3	B, G, P	

The “Type” specified in Table 2-4 relates to the sysplex target environment—**B** represents a BronzePlex, **G** represents a GoldPlex, and **P** represents a PlatinumPlex.

Notes indicated in Table 2-4 are described below:

1. In the SFM policy, you can have a default definition (by specifying SYSTEM NAME(*)), and the values specified on that definition will be applied to any system that is not explicitly defined. However, we recommend that there is an explicit definition for each system in the sysplex. Depending on your target environment, you should update the SFM policy with an entry for the incoming system, specifying the WEIGHT and ISOLATETIME values. For more information on using and setting up SFM, refer to *z/OS MVS Setting Up a Sysplex,* SA22-7625.
2. Depending on your target environment (BronzePlex, GoldPlex, or PlatinumPlex), you may wish to change the setting of your CONNFAIL parameter.
3. SFM supports the movement of processor storage from one named LPAR to another named LPAR in case of a failure. Depending on your target environment (BronzePlex, GoldPlex, or PlatinumPlex), you may wish to change the action that SFM takes when a system fails.

2.6 Automatic Restart Manager

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

The goals of SFM and ARM are complementary. While SFM helps improve sysplex availability by removing dead systems in a timely manner, ARM keeps specific work in the sysplex running. If a job or task fails, ARM restarts it on the same system it was running on at the time of the failure. If a system fails, ARM restarts the work on other systems in the sysplex that are in the same JES2 MAS or JES3 complex; this is called a cross-system restart.

In the ARM policy, you identify, among other things, the ARM element name of the job or STC that you want ARM to restart, and which systems it can be restarted on.

If your target environment is a BronzePlex, we assume that the incoming system will not be in the same MAS as the other systems in the sysplex. In this case, the work from any of the existing systems in the target sysplex cannot be restarted on the incoming system, and the the jobs from the incoming system cannot be restarted on the other systems in the target sysplex.

In your ARM policy, you can simply merge the two ARM policies together because ARM will only attempt to restart an element on a system in the same JES2 MAS/JES3 Complex as the failed system. You can continue to specify RESTART_GROUP(*) to specify defaults for all restart groups in the sysplex or TARGET_SYSTEM(*) for the restart group associated with the element(s) from the incoming system as long as these statements name systems that are in the same JES2 MAS/JES3 Complex as the failing system. If you use this statement and a system fails, and the ARM policy indicates that the element should be restarted, ARM will attempt to restart it on any other system in the same MAS/Complex. Also, remember that the ARM policy works off the element name of jobs and started tasks, which must be unique within the sysplex, so even if you have a started task with the same name in more than one system, you can still define them in the ARM policy as long as the application generates a unique element name.

If your target environment is a GoldPlex, we assume once again that you are not sharing the JES spool between the incoming system and the other systems in the target sysplex. Therefore, the same considerations apply for a GoldPlex as for a BronzePlex.

Finally, if your target environment is a PlatinumPlex, which infers full workload balancing, you can potentially restart any piece of work on any system in the sysplex. In this case, you need to review your existing ARM definitions to decide if the incoming system should be added to the TARGET_SYSTEM list for the existing ARM elements. In addition, you should define the work in the incoming system that you want ARM to manage, and decide which of the other systems in the sysplex are suitable candidates to restart that work following a failure.

2.6.1 Considerations for merging ARM

This section contains, in checklist format, a list of items that must be considered when you decide to merge two or more ARM policies, or to move a new system into an existing ARM environment.

Table 2-5 Considerations for merging ARM

Consideration	Note	Type	Done
Review ARM definitions and target systems	**1**	**B, G, P**	
Check if ARM is defined to the automation subsystem	**2**	**B, G, P**	

The “Type” specified in Table 2-5 relates to the sysplex target environment—**B** represents a BronzePlex, **G** represents a GoldPlex, and **P** represents a PlatinumPlex.

Notes indicated in Table 2-5 are described below:

1. Bearing in mind that ARM will only restart work on systems in the same JES MAS, for each element (batch job or started task) of ARM from the incoming system, you need to:
   – Determine which batch jobs and started tasks will be using ARM for recovery purposes. For the IBM products that use ARM, read their documentation for any policy considerations.

- Check for interdependencies between work—that is, elements that need to run on the same system (RESTART_GROUP).

  Note that any elements that are not explicitly assigned to a restart group become part of the restart group named DEFAULT. Thus, if these elements are restarted, they are restarted on the same system.
- Determine if there is an order in which MVS should restart these elements —if any elements in the restart group are dependent upon other elements being restarted and ready first (RESTART_ORDER).
- Determine if the elements in a restart group need to be restarted at specific intervals (RESTART_PACING).
- Determine if the element should be restarted when only the element fails, or when either the element or the system fails (TERMTYPE).
- Determine whether specific JCL or command text is required to restart an element (RESTART_METHOD).
- Determine if a minimum amount of CSA/ECSA is needed on the system where the elements in a restart group are to be restarted (FREE_CSA).
- Determine if you want the elements in the restart group to be restarted on a specific system (TARGET_SYSTEM). Consider requirements like:
  - Which systems have access to the ARM CDSs.
  - Which systems run within the same JES XCF group.
  - The workloads of the systems in the sysplex, and how they might be affected if these jobs were restarted.
  - CPU capacity needed.
  - DASD requirements.
  - Which systems have the class of initiator required by batch jobs that might need to be restarted on another system.
- Determine if an element should be restarted and, if so, how many times it should be restarted within a given interval (RESTART_ATTEMPTS). If an element should not be restarted, set RESTART_ATTEMPTS to 0.
- Determine how long MVS should wait for an element to re-register once it has been restarted (RESTART_TIMEOUT).
- Determine how long MVS should wait for an element to indicate it is ready to work after it has been restarted (READY_TIMEOUT).

2. Most automation products provide the ability to specify whether a given piece or work will be managed by ARM or by the automation product. If you will be adding products to ARM as a part of the merge process, you must update any affected automation products.

2.7 Tools and documentation

In this section we provide the list of tools and documentation mentioned in this chapter that may help you complete this task.

The following tools may be helpful in the planning and execution of the system merge:

- The CFSizer wizard is available on the Internet at:

  http://www.ibm.com/servers/eserver/zseries/cfsizer/

- Coupling Facility Level (CFLEVEL) information is available on the Internet at:

  http://www.ibm.com/servers/eserver/zseries/pso/cftable.html

- The SYS1.SAMPLIB library contains a number of sample jobs to allocate the various CDSs and create the policy for those CDSs. The member names all start with IXC, and the ones that create the policies end with a P.

The following documentation may be helpful as you merge the entities discussed in this chapter:

- *OS/390 MVS Parallel Sysplex Capacity Planning,* SG24-4680
- *OS/390 Parallel Sysplex Configuration Volume 2: Cookbook,* SG24-5638
- *WSC Flash 10011, entitled "XCF Performance Considerations"*
- *WSC Flash 98029, entitled "Parallel Sysplex Configuration Planning for Availability"*
- *z/OS MVS Setting Up a Sysplex,* SA22-7625

3

System Logger considerations

This chapter discusses the following aspects of moving a system that is using the System Logger into a sysplex where the System Logger is already being used by the existing systems:

- Is it necessary to merge System Loggers, or is it possible to have more than one Logger environment in a single sysplex?
- A checklist of things to consider should you decide to merge the Loggers.
- A methodology for doing the merge.
- A list of tools and documentation that can assist with this exercise.

3.1 One or more LOGRplexes

It is *not* possible to have more than one System Logger environment in a single sysplex. Only one primary LOGR Couple Data Set (CDS) may exist for any sysplex, regardless of the number or type of log streams, or the applications which use those log streams. In addition, unlike the other sysplex CDSs, you can only have one policy for any given LOGR CDS. A *LOGRplex* is the set of systems that all share the same LOGR CDS, and, as you can only have one LOGR CDS per sysplex, the scope of the LOGRplex is therefore the same as the scope of the sysplex.

As discussed in 1.2, “Starting and ending points” on page 2, there are three potential target environments: the BronzePlex, the GoldPlex and the PlatinumPlex. Even though you can only have one LOGRplex per sysplex, there are different considerations for the various target environments and these are discussed in the following sections.

3.2 Considerations for merging System Loggers

This section contains, in checklist format, a list of things that must be considered when you merge two or more System Logger environments. However, before discussing the specific considerations for the different users of System Logger, we will provide a little background information about how Logger works.

3.2.1 System Logger fundamentals

The System Logger supports two types of log streams:

- Those that are allocated in a Coupling Facility
- Those that use staging data sets on DASD to hold the data—these are referred to as DASDONLY log streams

CF log streams can be grouped so that more than one log stream resides in a Logger structure in the CF. These log streams can have more than one system connected to them at a time. DASDONLY log streams are not grouped—each log stream is only on DASD, in its own staging data set and does not use the CF. In addition, DASDONLY log streams only support connections from a single system. Generally speaking, when planning for merging sysplexes, the considerations are the same regardless of whether the log streams are DASDONLY or reside in CF structures. Where there are specific considerations for DASDONLY, we will point that out where applicable.

The next thing to address is the LOGR CDS. Like all the other sysplex CDSs, the LOGR CDS contains the LOGR policy that you define using the IXCMIAPU utility. And, like the other sysplex CDSs, it also contains transient information about the status of the LOGRplex. However, it differs from the other sysplex CDSs in that the LOGR CDS also contains *persistent* information—specifically, information about the staging and offload data sets. It is the presence of this information in the LOGR CDS, and the fact that there is no capability to merge that information, that makes System Logger environments a bit more complex to merge than the other sysplex CDSs.

Because you cannot carry over the persistent information from one LOGR CDS to another, you must ensure that all the Logger applications can restart successfully after the merge without requiring any of the information that resided in the incoming system’s log streams immediately prior to the merge. Information about how to achieve this is provided in Table 3-1 on page 41 and the associated notes.

Another point about Logger that you must understand is how the offload process works for log streams in the CF. In the LOGR policy, when you define each log stream, you specify which CF structure that log stream will reside in. You can have many log streams per structure. When a log stream reaches the high threshold specified in the LOGR policy, a notification is sent to all the MVS images connected to that log stream. The connected images will then decide between themselves which one will do the offload; you cannot control which system does the offload.

Furthermore, if a system fails, and that system was the last one connected to a given log stream, any system that is connected to the *structure* that the log stream resided in becomes a candidate to offload that log stream to DASD—this process is called Peer Recovery and is intended to ensure that you are not left with just a single copy of the log stream data in a volatile medium. For example, if system MVSA fails and it was the only system connected to a particular log stream, any of the systems that are connected to the structure that log stream resided in may be the one that will do the offload. This design is important as it affects how you will design your Logger environment, depending on whether your target environment is to be a BronzePlex, a GoldPlex, or a PlatinumPlex.

Note: The concern about which system does the offload for a log stream only relates to log streams in CF structures. Log streams on DASD (DASDONLY) are only offloaded by the system they are connected to, so the situation where the offload data sets are not accessible from one of the connectors does not arise.

The final thing we want to discuss is the correlation between the LOGR CDS and the staging and offload data sets. The LOGR CDS contains information about all the staging and offload data sets for each log stream. This information is maintained by Logger itself as it manages (allocates, extends, and deletes) the data sets, and there is no way to alter this information manually. So, for example, if you delete an offload data set manually, Logger will not be aware of this until it tries to use that data set and encounters an error. Equally, if you have an existing offload data set that is not known to the LOGR CDS (for example, if you have a brand new LOGR CDS), there is no way to import information about that data set into Logger.

So, consider what will happen when you merge the incoming system into the target sysplex. If you did not delete the staging and offload data sets before the merge, and if you are using the same user catalog, high level qualifier, and DASD volume pool after the merge, Logger will attempt to allocate new staging and offload data sets. However, it will encounter errors, because the data set name that it is trying to use is already defined in the user catalog and on the DASD volumes. The best way to address this situation varies depending on the target environment, and is discussed in the notes attached to Table 3-1 on page 41.

3.2.2 Considerations for different target environments

Regardless of your target environment, you will need to address the issue that any data that was resident in the incoming system's log streams immediately prior to the merge will no longer be accessible after the merge. How you address this issue depends on which applications are using Logger in your site and is discussed in Table 3-1 on page 41 and the associated notes.

There are also some Logger considerations that *are* impacted by your target environment.

In a BronzePlex, there is no sharing of user DASD between the incoming system and the target sysplex. Now, what would happen if a particular log stream (OPERLOG, for example) is connected to the incoming system and also to one or more systems in the target sysplex and the log stream reaches the high threshold? All the connected systems will compete to do the offload—let us assume that the incoming system wins on this occasion. The incoming system

will allocate an offload data set on one of *its* DASD and offload the data. Now, what happens if one of the systems in the target sysplex needs to retrieve data from that log stream, and the required data has been offloaded by the incoming system? The offload data set will not be accessible to the target sysplex system and the request will fail. Therefore, in a BronzePlex, you *must* ensure that none of the log streams are connected to by systems in both subplexes.

Furthermore, consider what happens during Peer Recovery. In this case, you can end up with log stream data on an inaccessible device because the *structure* was shared between the incoming system and one or more of the systems in the target sysplex. So, in a BronzePlex, not only should you not share *log streams* between the incoming system and the target sysplex systems, you should also not share any Logger *structures* in this manner either.

Similarly, in a GoldPlex, you would not normally share user catalogs or user DASD volumes. So, once again, you must ensure that the Logger structures are only connected to by systems in one of the subplexes. However, what if you decide to share the user catalog containing the Logger offload data sets and the DASD volumes containing those data sets—does this remove this requirement? It depends on how you manage the offload data sets. If you use DFSMShsm to migrate the offload data sets, then you have to be sure that any system in the LOGRplex can recall an offload data set that has been migrated; and this means that all the systems in the LOGRplex are in the same HSMplex, which effectively means that you have a PlatinumPlex (which we will discuss next). However, if you do *not* have a single HSMplex, then you cannot use DFSMShsm to manage the Logger offload data sets.

Finally, we have the PlatinumPlex. In a PlatinumPlex, everything is shared—all the user DASD, DFSMShsm, user catalogs, and so on. In this case, you still have the problem of ensuring that none of the data that is currently in the log streams of the incoming system will be required after the merge; however, you do not have any restrictions about which systems can connect to any of the Logger structures or log streams.

3.2.3 Considerations for the merge

Since there is no capability for merging multiple LOGR CDSs, the only way to proceed is to move the Logger policy definitions from the incoming system into the target sysplex LOGR CDS, and to ensure that none of the information in the incoming system log streams will be required after the merge. The checklist in this section discusses which actions must be taken for each application before moving to a new Logger environment.

In addition the log stream attributes from the incoming system must be considered, along with the total number of log streams and structures that will be defined (whether active or not) in the target sysplex LOGR CDS. In particular, the formatting of the LOGR CDS must contain ample room for the:

- Total number of log streams defined in the sysplex. This is specified via:
  - ITEM NAME(LSR) NUMBER( )
- Total number of structures defined in the sysplex. This is specified via:
  - ITEM NAME(LSTRR) NUMBER( )
- Total number of log streams that will have more than 168 offload data sets. For every such log stream, you must add 1 to the NUMBER specified via:
  - ITEM NAME(DSEXTENT) NUMBER( )

If the values specified in the existing target sysplex LOGR CDS are not sufficient to cater for the target environment, a new LOGR CDS must be defined, and brought into service.

Table 3-1 Considerations for merging System Logger

Consideration	Note	Type	Done?
Which Logger structure will each of the log streams be placed in?	**1**	**B,G,P**	
Do the staging and offload data set names match the requirements of your target environment?	**2**	**B,G**	
How to handle staging and offload data sets from the incoming system?	**3**	**B,G,P**	
Preparing the OPERLOG log stream for the merge	**4**	**B,G,P**	
Preparing the LOGREC log stream for the merge	**5**	**B,G,P**	
Preparing the IMS log streams for the merge	**6**	**B,G,P**	
Preparing CICS log streams for the merge	**7**	**B,G,P**	
Preparing the RRS log streams for the merge	**8**	**B,G,P**	
Preparing WebSphere® log streams for the merge	**9**	**B,G,P**	

The “Type” specified in Table 3-1 relates to the sysplex target environment—**B** represents a BronzePlex, **G** represents a GoldPlex, and **P** represents a PlatinumPlex.

Notes indicated in Table 3-1 are described below:

1. When you move to a single LOGR policy, you need to decide which Logger structure each log stream will be placed in after the merge. The simplest thing, of course, would be to leave all the log streams in the same structures they are in prior to the merge, and this is a viable strategy. However, you must ensure that there are no duplicate log stream or Logger structure names between the incoming system and the target sysplex. Any duplicate log stream names will need to be addressed (this will be discussed in subsequent notes). Duplicate Logger structure names will also need to be addressed. Changing the Logger structure names consists of:
   - Adding the new structure names to the CFRM policy.
   - Updating the LOGR policy to define the new structure names (on the DEFINE STRUCTURE statements) and updating the log stream definitions to assign a new structure to the affected log streams.

   Note: This consideration does not apply to DASDONLY log streams.

2. There is a general consideration that applies to all of the applications that use log streams (either CF log streams *or* DASDONLY), and it is related to the offload data sets. Any Logger subsystem that has a connection to a Logger structure can potentially offload a log stream from that structure to an offload data set (allocating a new offload data set in the process, if necessary). If the target environment is a PlatinumPlex, this does not represent a problem, as the SMS configuration, DFSMShsm environment, user DASD, and all the catalogs will be shared by all the systems in the sysplex. However, if the target environment is a BronzePlex or a GoldPlex, some planning is required when defining the log stream.

   To avoid problems with potential duplicate data set names or duplicate Master Catalog aliases, you should use different high level qualifiers for the offload data sets associated with each subplex. The default HLQ for offload data sets is IXGLOGR; however, you can override this on the log stream definition using the HLQ (or, in z/OS 1.3 and later, the EHLQ) keyword. This keyword also controls the HLQ of the staging data sets (if any) associated with each log stream. Using different HLQs for each subplex means that you

can use different catalogs for the Logger data sets, have different (non-clashing) RACF profiles, and different SMS constructs in each subplex. Avoiding duplication now means that should you decide to move to a PlatinumPlex in the future, that move will be a little bit easier.

3. Earlier in this chapter we discussed the relationship between the LOGR CDS, the user catalog(s) associated with the staging and offload data sets, and the volumes containing those data sets and the problems that can arise should these get out of sync with each other. The options for addressing this situation depends on your target environment.

   If your target environment is a BronzePlex or a GoldPlex, the incoming system will probably be using the same user catalog and same DASD volumes after the merge as it did prior to the merge. This means that the old staging and offload data sets will still exist on the DASD and be cataloged in the user catalog. As soon as Logger on the incoming system tries to allocate new versions of these data sets after the merge, you will encounter an error. There are two ways to address this:

   a. You can delete all the old offload and staging data sets on the incoming system after you have stopped all the Logger applications and before you shut down that system just prior to the merge. You do this by deleting the log stream definitions from the Logger policy on that system. This will ensure that you do not encounter duplicate data set name problems when you start up again after the merge. However, it means that the log streams will need to be redefined if you have to fall back.

   b. You can change the HLQ associated with the log streams from the incoming system when defining those log streams in the target sysplex LOGR CDS. In this case, you do not have to delete the log stream definitions from the incoming system (making fallback easier, should that be necessary) as the new offload and staging data sets will have a different high level qualifier than the existing data sets.

   If your target environment is a PlatinumPlex, the incoming system will probably switch to using the user catalog that is currently used for the offload and staging data sets on the target sysplex systems. This means that you would not have the problem of duplicate user catalog entries; however, you would need to decide how to handle the existing staging and offload data sets. Once again, you have two options:

   a. You can delete all the old offload and staging data sets on the incoming system after you have stopped all the Logger applications and before you shut down that system just prior to the merge. You do this by deleting the log stream definitions from the Logger policy on that system. You would do this if the DASD volumes containing the offload and staging data sets will be accessible in the new environment.

   b. You can leave the old offload and staging data sets allocated, but make the volume containing those data sets inaccessible to the new environment. This means that those data sets are still available should you need to fall back. However, it may be complex to arrange if the offload and staging data sets are on the same volumes as other data sets that you wish to carry forward.

4. The operations log (OPERLOG) is a log stream with a fixed name (SYSPLEX.OPERLOG) that uses the System Logger to record and merge communications (messages) 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 OPERLOG log stream. Because the OPERLOG log stream has a fixed name, there can only be one OPERLOG within a sysplex—that is, you cannot have SYSA and SYSB share one OPERLOG, and have SYSC with its own, different, OPERLOG.

The operations log is operationally independent of the system log. An installation can choose to run with either or both of the logs. If you choose to use the operations log as a replacement for SYSLOG, you can prevent the future use of SYSLOG; once the operations log is started with the SYSLOG not active, enter the WRITELOG CLOSE command.

Although the operations log is sysplex in scope, the commands that control its status and the initialization parameter that activates it have a system scope, meaning that a failure in operations log processing on one system will not have any direct effect on the other systems in the sysplex. You can set up the operations log to receive records from an entire sysplex or from only a subset of the systems, depending on the needs of the installation.

Depending on the type of the target environment, some configuration considerations apply: in a BronzePlex or a GoldPlex where there is need for separation of data between the two subplexes, OPERLOG should only be active for the images that belong to one subplex, with the other images continuing to operate through SYSLOG.

On the other hand, in a BronzePlex/GoldPlex where there is an operational need to have all the messages accumulated into a single OPERLOG, additional planning is required to determine how to handle the offload of this log stream; since offload can take place on any image connected to that structure, the catalog that the offload data sets are cataloged in must be shared, the pool of volumes that the offload data sets will reside on must be shared, the SMS constructs in each subplex should be identical in their handling of Logger offload and staging data sets, and it is not possible to use DFSMShsm to migrate them. In a PlatinumPlex, OPERLOG should collect data from all the images and there should no additional considerations for the offload data sets, since all the storage environment should be shared.

In light of the fact that the OPERLOG data, as is also true with SYSLOG records, is not used for backup, recovery, or repair for any applications or programs, the retention or transportation of OPERLOG data is not critical as it would be for an application which depended upon its log stream data for recovery, such as CICS.

Initially the OPERLOG messages which live in the log stream are in MDB format. Typically the IEAMDBLG sample program (or a modified version thereof) is run to move the OPERLOG messages out of the log stream into “flat files” where they can be archived in a SYSLOG format. Because of this capability, the data in the OPERLOG log stream of the incoming system does not need to be lost as part of the merge—however, that data will now need to be accessed from outside Logger.

5. Before discussing the LOGREC log stream, we wish to point out that IBM recommends that you IPL with a LOGREC data set initialized by IFCDIP00. If you do not IPL with a LOGREC data set, you cannot subsequently change the LOGREC recording medium from LOGSTREAM to DATASET using the SETLOGRC command. When planning for the merge, ensure that you have taken this action so that in the event that you experience any problems re-establishing the LOGREC log stream, you will at least have a medium to which you can fall back.

   The LOGREC log stream provides an alternative, shared, medium for collecting LOGREC data. Like OPERLOG, the use of the LOGREC log stream can be controlled on a system-by-system basis. So, none, some, or all of the systems in the sysplex can be set up to use the LOGREC log stream. Similarly, like OPERLOG, the LOGREC log stream has a fixed name, SYSPLEX.LOGREC.ALLRECS, so you can only have one LOGREC log stream per sysplex.

When you merge the sysplexes, there is no need to carry forward the LOGREC data. There are two reasons for this:

a. The first is that the old LOGREC data may be archived using the IFCEREP1 program to move the log stream data to a flat file. Once this is done, you have (and can retain) this data from the old sysplex for as long as necessary.

b. The other reason is that the target sysplex may have no knowledge of the previous systems and cannot be configured to be able to read the old sysplex's log stream LOGREC records.

Because the LOGREC log stream is like the OPERLOG log stream in terms of having a fixed name, the same considerations apply for BronzePlex, GoldPlex, and PlatinumPlex environments as we described above for the OPERLOG log stream.

6. The IMS Common Queue Server (CQS) uses the System Logger to record information necessary for CQS to recover its structures and restart following a failure. CQS writes log records for the CF list structures to log streams (the Message Queue structure and its optional associated overflow structure to one log stream, and the optional EMH Queue and EMH Overflow structures to a second log stream). The log streams are shared by all the CQS address spaces that share the structures. The System Logger therefore provides a merged log for all CQS address spaces that are sharing queues.

   Merging the System Logger environments will mean that any information in the CQS log streams from the incoming system will no longer be accessible after the merge. This means that all the IMSs on the incoming system will need to be shut down in such a manner that they when they are started up again (after the merge) they will not need to get any information from the log stream. You should work with your IMS system programmers to ensure that whatever process they normally use to shut down IMS prior to an IMS and CQS cold start is completed successfully. Moving to a new Logger environment is effectively no different, from a shared queue perspective, than a normal cold start of CQS.

   As long as you have removed the need for IMS to require log stream data to start up, the only consideration to be taken into account is to move all the CQS log stream definitions into the LOGR policy of the target sysplex.

   Note: We are assuming that as part of the merge you are *not* merging the IMS from the incoming system into the shared queue group from the target sysplex. If your intent is to place all the IMSs in a single shared queues group, such a change would normally not take place at the same time as the merge of the sysplexes.

7. CICS uses System Logger services to provide logging in the following areas: *backward recovery (backout)*, *forward recovery,* and *user journals*.

   Backward recovery, or backout, is a way of undoing changes made to resources such as files or databases. Backout is one of the fundamental recovery mechanisms of CICS. It relies on recovery information recorded while CICS and its transactions are running normally. Before a change is made to a resource, the recovery information for backout, in the form of a before-image, is recorded in the CICS system log (also known as DFHLOG). A before-image is a record of what the resource was like before the change. These before-images are used by CICS to perform backout in two situations:

   – In the event of failure of an individual in-flight transaction, which CICS backs out dynamically at the time of failure (dynamic transaction backout), or,

   – In the event of an emergency restart, where CICS backs out all those transactions that were in-flight at the time of the CICS failure (emergency restart backout)

Each CICS region has its own system log written to a unique System Logger log stream. Since the CICS system log is intended for use only for recovery purposes during dynamic transaction backout or during emergency restart, there is no need to keep the logged data if the CICS region has been shut down correctly. If CICS regions are moved from one Logger environment to a new one, there is no need to bring the CICS system log data as long as you can ensure that CICS will not need to do any sort of recovery when it starts in the new environment.

Note: What do we mean by shutting CICS down "correctly"? The objective is to ensure that there were no in-flight transactions when CICS stops. Normally these in-flight transactions would be handled when CICS restarts because it is able to obtain information about these in-flight transactions from the system log log stream. However, we are going to be discarding the system log log stream, so you have to ensure that there are no in-flight transactions.

The way to do this is to use CEMT to list all active tasks and cancel any that have not completed. When you cancel the task, CICS will back out the associated transaction, ensuring that no transactions are only half done. Once all active tasks have been addressed, CICS will then shut down cleanly and you can do an INITIAL start when CICS is started after the merge.

Forward recovery logs are used when some types of data set failure cannot be corrected by backward recovery; for example, failures that cause physical damage to a database or data set. To enable recovery in these situations you must take the following actions:

a. Ensure that a backup copy of the data set has been taken at regular intervals.
b. Set up the CICS regions so that they record an after-image of every change to the data set in the forward recovery log (a log stream managed by the System Logger).
c. After the failure, restore the most recent backup copy of the failed data set, and use the information recorded in the forward recovery log to update the data set with all the changes that have occurred since the backup copy was taken.

The forward recovery log data usually does not reside in the log stream for a long time. Normally an installation will use a forward recovery product, such as CICSVR, to manage the log stream. These products extract the data from the log stream, synchronized with the data set's backup copy, and store the forward recovery data in an archive file. After they extract the required records from the log stream, they tell Logger to delete those records. Should a forward recovery be required, the product will use the information from the archive file or the forward recovery log stream, depending on where the required records reside at that time.

If you backup all the CICS files that support forward recovery immediately after you shut down CICS in preparation for the merge, all the required records will be moved from the forward recovery log stream to the archive files, meaning that there is no required data left in the log stream. If, for whatever reason, you need to restore one of the CICS data sets before you restart CICS, all the required information will be available through the backups and archive files.

*User journal*s are not usually kept for a long time in the log stream, and the data in the log stream is not required again by CICS. They are usually offloaded using the DFHJUP utility in order to be accessed by user applications, vendor products, and to be preserved by being archived. For this reason, as long as you extract whatever data you need from the log stream before shutting the incoming system down for the merge, the user journal log streams will not contain any data that is required after the merge.

Assuming that you will not be merging CICS Logger structures as part of the merge, the considerations for the CICS log streams are the same regardless of whether the target environment is a BronzePlex, a GoldPlex, or a PlatinumPlex.

After the merge, all the CICS regions on the incoming system should be started using an INITIAL restart.

8. Depending on the target environment, you can have different RRS configurations. In a BronzePlex or a GoldPlex, you will wish to keep the RRS environment for the incoming system separate from that of the target sysplex. On the other hand, in a PlatinumPlex, you should have a single RRS environment across the entire sysplex. However, even if you will keep two separate RRS environments, there is still no way to carry the data in the incoming system's log streams over into the target environment. Therefore, the following considerations apply regardless of the target environment type.

   RRS uses five log streams, each of which is shared by all the systems in the RRSplex. An RRSplex is the set of systems that are connected to the same RRS log streams. The RRS term for an RRSplex is "logging group". So, for example, you can have production systems sharing one set of RRS log streams and test systems sharing a different set of RRS log streams in the same sysplex. A logging group is a group of systems that share an RRS workload. The default log group name is the sysplex name; however, this can be overridden via the GNAME parameter in the RRS startup JCL. The GNAME is used as the second qualifier in the log stream name.

   Note: Because there is only one RRS subsystem per system, each system can only be in a single RRSplex.

   The five log streams are:

   – ATR.gname.ARCHIVE. This log stream contains information about completed Units of Recovery (URs). This log stream is optional and should only be allocated if you have a tool to help you analyze the information in the log stream.
   – ATR.gname.RM.DATA. This log stream contains information about Resource Managers that are using RRS services.
   – ATR.gname.MAIN.UR. This log stream contains information about active URs. RRS periodically moves information about delayed URs into the DELAYED.UR log stream.
   – ATR.gname.DELAYED.UR. This log stream contains the information about active but delayed URs that have been moved from the MAIN.UR log stream.
   – ATR.gname.RESTART. This log stream contains information about incomplete URs that is required during restart.

   When all the Resource Managers shut down in a clean way (that is, with no in-flight URs), there is no information that is needed to be kept in the RRS log stream. Therefore, in order to be able to restart RRS after the merge with a new set of empty log streams, you need to ensure that all the Resource Managers have shut down cleanly.

   Note: IMS is one of the potential users of RRS. However, if RRS services are not required by IMS (that is, you are not using ODBA or Protected Conversations), you can use the RRS=N parameter (added to IMS V7 by APAR PQ62874) to stop IMS from connecting to RRS.

   If you *do* require IMS use of RRS, the /DIS UOR IMS command displays any in-doubt UORs involving RRS.

If your target environment is a BronzePlex or a GoldPlex, you should modify the RRS startup JCL on the incoming system to specify a GNAME other than the sysplex name. If you can set GNAME to be the same as the sysplex name of the incoming system, you can copy the Logger policy definitions for the RRS log streams exactly as they exist in the incoming system.

If your target environment is a PlatinumPlex, all of the systems *must* be in the same RRSplex. In this case you may need to review the log stream sizes as currently defined in the target sysplex to ensure they are large enough for the additional load that may be generated by the incoming system.

9. WebSphere for z/OS uses a Logger log stream to save error information when WebSphere for z/OS detects an unexpected condition or failure within its own code, such as:
   - Assertion failures
   - Unrecoverable error conditions
   - Vital resource failures, such as memory
   - Operating system exceptions
   - Programming defects in WebSphere for z/OS code

   The name of the log stream is defined by the installation, and you can either have a separate log stream per WebSphere Application Server (WAS) server, or you can share the log stream between multiple servers. If you have a separate log stream for each server, the log stream can be a CF log stream or it can be a DASDONLY one.

   Regardless of the log stream location, or the target environment type, because the data in the log stream is not required for anything other than problem determination, it is not necessary to preserve the data after a clean shutdown of the server. You can use the BBORBLOG REXX exec, which allows you to browse the error log stream to ensure that you have extracted all the information you require about the errors before the log stream information is discarded.

3.3 Methodology for merging System Loggers

As discussed previously, there is no tool to merge the contents of multiple LOGR CDSs. However, as long as you prepare a comprehensive plan, it should be possible to complete the merge with no Logger-related problems.

In this section, we are providing a generic set of steps that can be applied to all the Logger applications. The details of each step may vary from one exploiter to another, however, those details have already been discussed.

Step 1 - Compare Logger policy definitions

The very first step is to identify the log streams defined in the incoming system and target sysplex Logger policies, checking for duplicate log stream names, and identifying any log streams that will be connected to both the incoming system and one or more systems from the target sysplex.

For log streams that will have more connectors after the merge, review the log stream definition to ensure there is enough room in the offload data sets for the increased volume of data, and check the associated structure definitions to ensure they are large enough to cater for the additional load.

At the end of this step you should have decided what you are going to call all the log streams and Logger structures, which structure each log stream will reside in, and any changes that may be required to existing structure and log stream definitions.

Step 2 - Add definitions to target sysplex Logger and CFRM policies

In order to be able to restart successfully after the merge, the log streams and Logger structures from the incoming system must be defined to the target sysplex Logger and CFRM policies.

For the structures, you need to decide if you will continue to use the same structure names in the new merged environment. The whole issue of merging CFRM definitions is covered in detail in 2.4, “Coupling Facility Resource Management” on page 27. Here, we will simply say that you must ensure that the CFRM policy contains definitions for any structures referenced in the Logger policy.

In the Logger policy, you must ensure that you have defined all the structures that are referenced by any of the log streams. In addition, you must consider which log streams will reside in the same structure—this is especially important in a BronzePlex or a GoldPlex environment. You may also wish to change the HLQ associated with a log stream’s staging and offload data sets—once again, you are more likely to make this change in a BronzePlex or a GoldPlex environment

Step 3 - Shutting down before the merge

This will be carried out as you shut down all users of Logger on the incoming system just before the IPL to merge the sysplexes.

You should shut down all of the Logger exploiters cleanly, ensuring that there are no in-flight transactions still running when you stop the subsystems.

For OPERLOG and LOGREC, you can issue commands to stop the use of those log streams, then run the required batch jobs to archive whatever data you require out of those log streams.

For CICS, you must ensure that all transactions are either completed or cancelled before CICS stops. The objective is to ensure that there are no started, but not ended, transactions in CICS when the region is shut down. After you shut down all the CICS regions, take backups of all the CICS data sets that support forward recovery, and ensure that all the forward recovery data has been archived from the forward recovery log stream(s) by the forward recovery products.

For IMS, you need to decide what is to be done with outstanding messages at the time of cutover:

- Process all of them before shutdown by stopping IMS with the PURGE option. This should cause IMS to process all transactions that can be processed and deliver all messages that can be delivered. If IMS has not stopped after some period of time, you can use the IMS /DISPLAY SHUTDOWN STATUS command to display regions that are still executing and nodes and lines that are still sending (space) or receiving data. Output messages that IMS cannot deliver and input messages that cannot be processed will be left on the queue.
- You can use a message requeuer product (such as Message Requeuer from IBM) to unload the queue that you are abandoning and load all those messages onto the new queue after the merge.
- You can abandon the remaining messages.

For WebSphere, stop the WebSphere address space then process any information you require from the WebSphere log stream.

As long as all the Resource Managers have been shut down cleanly, you will not require any of the information from the RRS log streams. To find out if any Resource Managers are still registered with RRS, or if there are any active URs, you can use the RRS panels—there is no way to display this information from the console. If there are any Resource Managers still registered or any in-flight URs, you need to resolve that situation before the shut down. Once you are happy that there are no in-flight URs, you do not have to do anything specific with RRS prior to shutting down the system.

Step 4: Restarting after the merge

Restarting the Logger applications after the merge should be relatively straightforward.

Whether you wish to use the OPERLOG and LOGREC log streams on the incoming system after the merge depends on your environment (BronzePlex, GoldPlex, or PlatinumPlex) and whether you are sharing the Logger offload DASD and user catalog. We recommend starting the system so that you are *not* using these log streams, then enabling them by command after the system has started successfully.

When you start the CICS regions for the first time after the IPL, you will need to do an INITIAL start on all the regions, indicating that they should initialize new system logs. Other than that, there are no special considerations for CICS.

When you start IMS, you should do a cold start of CQS (it should not be necessary to cold start IMS). When you bring up CQS and it connects to the new log stream, it will probably put out the message asking for a log token, to which you should reply “cold”.

RRS will start automatically after the IPL. If it finds that its log streams are not allocated, it will automatically do a cold start. As long as all the Resource Managers closed down cleanly, this should not present a problem.

Similarly, when WebSphere starts, it will allocate the new log streams and continue processing as normal.

Step 5 - Cleanup

The final step (unless you decided to do this prior to the shutdown of the incoming system) is to delete the old staging and offload data sets that were being used by the incoming system prior to the merge.

If you are using different High Level Qualifiers and are in a BronzePlex or a GoldPlex environment, the data sets should be available through the normal catalog search order and can be deleted using DELETE CLUSTER IDCAMS commands. If the data sets cannot be accessed through the normal catalog search order and are on SMS-managed volumes, you must use a DELETE NVR IDCAMS command to delete them (remember that you cannot use STEPCAT or JOBCAT with SMS-managed data sets).

3.4 Tools and documentation

In this section we provide information about documentation that may help you complete this task.

The primary reference for information about the System Logger is *z/OS MVS Setting Up a Sysplex,* SA22-7625. Specifically, the following sections in that book may be helpful:

- “Understand the Requirements for System Logger” in topic 9.4.1.
- “Develop a Naming Convention for System Logger Resources” in topic 9.4.4
- “Preparing to Use System Logger Applications” in topic 9.4.

- "Activate the LOGR Subsystem" in topic 9.4.12

Additional information about System Logger is available in Chapter 27, "Using System Logger Services" in *z/OS MVS Programming: Assembler Services Guide,* SA22-7605.

Information about IMS's use of System Logger services is available in:

- *IMS/ESA V6 Common Queue Server Guide and Reference,* SC26-9517
- *IMS Verson 7 Performance Monitoring and Tuning Update,* SG24-6404

Information about CICS's use of System Logger services is available in:

- *CICS TS Intallation Guide,* GC34-5985

Information about RRS's use of System Logger services is available in:

- *z/OS MVS Programming: Resource Recovery Services,* SA22-7616

4

WLM considerations

This chapter discusses the following aspects of moving a system that is in WLM Goal mode into a Parallel Sysplex where all of the existing systems are also in WLM Goal mode:

- Is it necessary to merge WLM environments, or is it possible to have more than one WLM environment in a single Parallel Sysplex?
- A checklist of things to consider should you decide to merge the WLMs.
- A methodology for doing the merge.
- A list of tools and documentation that can assist with this exercise.

4.1 One or more WLMplexes

It is *not* possible to have more than one WLMplex in a single sysplex. The definition of a WLMplex is the set of systems sharing a single WLM CDS and WLM service definition. The WLMs in the sysplex communicate via XCF, using an XCF group that has a fixed name (SYSWLM). Therefore, *all* the WLMs in the Parallel Sysplex will automatically communicate with each other. Part of the communication is to ensure that all the systems are using the same WLM CDS. If you IPL a system into the sysplex, and the COUPLExx member for that system points to WLM CDSs that are different to the ones currently in use in the sysplex, XCF ignores your definitions in COUPLExx, and automatically starts up using the CDSs that are being used by all the other members of the sysplex.

4.1.1 Compatibility mode considerations

If either the incoming system or all the systems in the target sysplex are currently in WLM compatibility mode, there may be nothing to do from a WLM perspective when moving the system into the sysplex. When WLM is in compatibility mode, each system has its own definitions (IPS and ICS members) and those definitions have no effect on the other members of the sysplex. Also, even in compatibility mode, WLM will use the service definition to classify enclaves into a PGN, to collect response time data for CICS or IMS into RPGNs, and for Scheduling Environments.

Remember that z/OS 1.2 is the last release to support WLM compatibility mode.

4.1.2 Configuration options

You will recall in 1.2, “Starting and ending points” on page 2 that we discussed the three potential target environments, namely BronzePlex, GoldPlex, and PlatinumPlex. In a BronzePlex, the minimum possible is shared, whereas in a PlatinumPlex, everything possible is shared.

Regardless of your target environment, however, there can only be a single WLM policy in effect for *all* the systems in the sysplex. Therefore, for WLM, you have no option but to merge the WLM policies from the incoming system and the target sysplex.

The difference between a BronzePlex/GoldPlex and a PlatinumPlex is that in a PlatinumPlex, all work can potentially run on all systems in the sysplex (so the work running in service class “A” in system SYSY is likely to be the same as the work running in that service class on system SYSZ), whereas in the other two environments, each piece of work can only potentially run on a subset of the systems (so the work running in service class “A” in system SYSR may be completely unrelated to the work running in that service class in system SYSS). You still need to create a single WLM service policy for the whole sysplex in both cases, but the setup of service classes and classification rules may be different depending on your target environment. This is discussed in more detail in the next section.

4.2 Merging WLMplexes

As the business of mergers and acquisitions continues to increase in the marketplace and the capacity of the newer processors continues to increase substantially, many customers are contemplating the consolidation of workloads onto one or more processors in one Parallel Sysplex. One of the challenges facing customers in WLM goal mode is how to incorporate the various service definitions from different Parallel Sysplexes into one service definition and set of policies which can then be installed and activated on one target sysplex.

It is important to understand that one of the principal philosophies of WLM/SRM management is based on workload sampling. Each item of work on the system is assigned to a service class period (SCP), and WLM performance information is maintained at the SCP level. The more work there is in an SCP, the greater the amount of sampling data available, giving WLM more information on which to make its decisions. If the number of active service class periods in a system can be limited to fewer than 25 or 30, then the more responsive WLM can be to the needs of the workloads.

It's also a good idea to try to limit the proportion of high importance periods. For example, if you have only 25 periods, but 20 of them are Importance 1, WLM may spend so many intervals fine-tuning the high importance work, that it may not be responsive enough to fluctuations in the work that is assigned a lower Importance.

At each 10-second interval, WLM in each system selects the SCP in the system that is most in need of help and takes one action on its behalf. The more SCPs you have that need assistance, the more 10-second cycles it takes to execute all the needed actions for these service classes. This is one of the reasons for the recommendation to limit the number of active SCPs on a system.

In addition to helping the SCPs on a system that are missing their goals, WLM will also attempt to help SCPs with a poor sysplex Performance Index (PI). Sometimes WLM will help an SCP on a system where that SCP is already meeting its goal, if it thinks that the adjustment will improve the sysplex PI of that SCP. This assumes that there is similar and related work running in that service class in every system in the sysplex.

If you have two workloads that run on different systems, and have *absolutely* nothing to do with each other, it may be wise to assign those workloads to different service classes. The reason being that WLM, in an attempt to improve the sysplex PI of that SCP, may help the service class on one system, to compensate for poor performance of that service class on the other system. If the two workloads are completely unrelated, this action may not have the desired effect.

Therefore, one of the first items to undertake in attempting to merge WLM workloads is to map out the service classes of the different policies into a matrix, ordering the service classes by importance as demonstrated in Table 4-2 on page 58 in order to merge (where appropriate) service classes and consequently decrease the number of service classes. This gives you a good overview of how the various workloads relate to one another with regards to their respective objectives.

Before you start making any changes to WLM, you should consider the following general recommendations for optimizing performance with WLM goal mode:

- Use percentile response time goals for service classes containing significant numbers of transactions. For service classes with smaller or inconsistent numbers of transactions (like a test CICS system, for example), velocity goals may be more effective.
- Review the service class period durations—too long can negate the benefit to shorter running transactions, while periods that are too short are ineffective if period switch occurs before any work has had a chance to complete.
- Ensure that your service definition coefficients make sense in your environment, but also remember that changing the coefficients can have a significant effect on how the system performs and may also affect chargeback algorithms.
- With velocity goals, the same work can achieve different velocities on processors of different speeds and different numbers of processors. Therefore, there may not be a single “right” velocity goal in a sysplex with different machines types. For work that requires velocity goals, don't try to be too precise in selecting a number. Small differences in velocity goals have little noticeable effect. For example, pick velocities that represent slow,

medium, and fast. These might be 20%, 40%, and 60%, respectively. The higher velocity goals are more sensitive to processor differences, so when selecting a fast velocity goal, select one that is attainable on the smaller processors. For more information on velocity, see the paper entitled “Velocity Goals: What You Don't Know Can Hurt You” on the WLM home page at:

http://www.ibm.com/servers/eserver/zseries/zos/wlm/

- Work in SYSSTC is not managed by WLM, so review the classification rules for tasks assigned to SYSSTC and make sure the work you classify there is necessary and appropriate for this service class. Make use of the SPM SYSTEM and SPM SYSSTC rules.
- Since the classification rules are an ordered list, review the order of work in the classification rules and place the more active and repetitive work higher in the list. Use Transaction Name Groups and Transaction Class Groups where possible.
- Use the new OS/390 2.10 enhanced classification qualifiers if it is necessary to treat similarly named work differently on different systems.
- Review the IEAOPT parms of all images to ensure consistency.
- Optimizing the number of logical CPUs benefits workloads that have large amounts of work done under single tasks, and minimizes LPAR overhead for all workloads. The WLM Vary CPU Management feature of IRD can help automate this process.
- Before and after making any WLM changes, extract RMF reports for both systems (incoming system and target sysplex). Verify the goals and redefine them, if necessary. It is very important to maintain the goals with the same availability and performance.

4.3 Considerations for merging WLMplexes

This section contains, in checklist format, a list of items that must be considered when you decide to merge two or more WLMplexes. We say WLMplexes rather than WLM CDSs, because there are more things to be considered than simply the contents of the CDSs.

Table 4-1 Considerations for merging WLMplexes

Consideration	Notes	Type	Done
Ensure the incoming system supports the Function level of the target sysplex WLM CDS.	1	B, G, P	
Check that the WLM CDSs are large enough to contain the merged definitions.	2	B, G, P	
Determine which services classes are required in the final configuration.	3	B, G, P	
Check for clashes in the WLM classification rules.	4	B, G, P	
Check for clashes in the WLM classification groups.	5	B, G, P	
Check for clashes in the WLM workload definitions.	6	B, G, P	
Check for clashes in the WLM report classes.	7	B, G, P	
Check if WLM service classes are used to report on workload goal attainment.	8	B, G, P	
Evaluate the impact of Resource Capping if used.	9	B, G, P	
Check if the WLM support feature of OPC is being used.	10	B, G, P	

Consideration	Notes	Type	Done
Check if Scheduling Environments are used in either environment.	**11**	**B, G, P**	
Check if Application Environments are used in either environment.	**12**	**B, G, P**	
Check usage of WLM-managed initiators.	**13**	**B, G, P**	
Check usage of response time rather than velocity goals for CICS and IMS.	**14**	**B, G, P**	
Check if either environment is using CICSplex/System Manager.	**15**	**B, G, P**	
Check if IRD is being exploited.	**16**	**B, G, P**	
Investigate use of dynamic PAV in either environment.	**17**	**B, G, P**	
Review the Performance Index (PI) for each service class.	**18**	**B, G, P**	
Investigate if the more granular control in classification rules introduced in OS/390 V2R10 will be required to classify work.	**19**	**B, G, P**	
Check if WLM constructs are used in reporting/chargeback jobs.	**20**	**B, G, P**	

The “Type” specified in Table 4-1 on page 54 relates to the sysplex target environment—**B** represents a BronzePlex, **G** represents a GoldPlex, and **P** represents a PlatinumPlex.

Notes indicated in Table 4-1 on page 54 are described the following:

1. There are two attributes of the target sysplex WLM CDS that you must consider: the Format Level, and the Function Level. Both can be determined by issuing a D WLM command. The format level controls which WLM functions can be used, and is determined by the level of OS/390 or z/OS that was used to allocate the CDS. At the time of writing, the highest Format Level is 3, and this is the Format Level that will be created by OS/390 2.4 and all subsequent releases. Considering how long this Format Level has been around, it is unlikely that anyone is still using a CDS with Format Level 1 or 2.

   The other thing to consider is the Function Level. The Function Level indicates which functions are actually being used in the CDS. For example, if you are using any of the WLM enhancements delivered in OS/390 2.10, the Function Level of the CDS would be 011. If you then start using the IRD capability to manage non-z/OS LPARs (a new function delivered in z/OS 1.2) the Function Level of the CDS would change to 013 as soon as the new policy is installed (Level 012 was reserved).

   The coexistence and toleration PTFs required in relation to the WLM CDS are documented in the chapter entitled “Workload Management Migration” in *z/OS MVS Planning: Workload Management,* GA22-7602. We recommend using the lowest functionality system to do maintenance, and to not enable any functionality not supported on the lowest-level image in the sysplex. This ensures that every system in the sysplex is able to activate a new policy.

2. You may need to resize the WLM CDSs for storing the service definition information. One of the inputs to the program that allocates the WLM CDS is the maximum number of systems in the sysplex. If the addition of the incoming system causes the previous MAXSYSTEM value to be exceeded, then you must allocate a new CDS (and don’t forget to allocate an alternate and spare at the same time) that reflects the new maximum number of systems in the sysplex. Note that if you used the WLM dialog to allocate the WLM CDSs, it will automatically set MAXSYSTEM to be 32, the highest value possible.

If you are running a sysplex with mixed release levels, you should format the WLM CDS from the highest level system. This allows you to use the latest level of the WLM application. You can continue to use the downlevel WLM application on downlevel systems provided that you do not attempt to exploit new function.

To allocate a new WLM CDS you can either use the facility provided in the WLM application, or you can run an XCF utility (IXCL1DSU). In each case, you need to estimate how many workload management objects you are storing on the WLM CDS. You must provide an approximate number of the following:

a. Policies in your service definition

b. Workloads in your service definition

c. Service classes in your service definition

We recommend using the WLM dialog to allocate the new data sets because the dialog will extract the values used to define the current data sets. If you don't use the dialog, then you should save the JCL that you use to allocate the data sets for the next time you need to make a change.

The values you define are converted to space requirements for the WLM CDSs being allocated. The total space is *not* strictly partitioned according to these values. For most of these values, you can consider the total space to be one large pool. If you specify 50 service classes, for instance, you are simply requesting that the space required to accommodate 50 service classes be added to the CDS—you have *not* limited yourself to 50 service classes in the service definition. Although note that if you *do* define more than 50 service classes, you will use up space that was allocated for something else.

Example 4-1 Allocate Couple Data Set for WLM using CDS values

```
File  Utilities  Notes  Options  Help
-------------------------------------------------------------------------------
!            Allocate couple data set for WLM using CDS values                !
! Command ===> ____________________________________________________________  !
!                                                                             !
! Sysplex name    ________  (Required)                                        !
! Data set name   ________________________________________________  (Required) !
!                                                                             !
!  ---------------------------------------------------------------------  !
!                                  |                                      !
! Size parameters                  |                                      !
! (optional):                      |  Storage parameters:                 !
!                                  |                                      !
! Service policies . . 10  (1-99)  |  Storage class  . . . ________        !
! Workloads  . . . . . 35   (1-999)|  Management class . . ________        !
! Service classes  . . 100  (1-999)|                                      !
! Application                      |  or                                  !
! environments . . . . 50   (1-999)|                                      !
! Scheduling                       |  Volume . . . . . . . ______         !
! environments . . . . 50   (1-999)|  Catalog data set?    _  (Y or N)    !
! SVDEF extensions . . 5     (0-8092)|                                     !
! SVDCR extensions . . 5     (0-8092)|                                     !
! SVAEA extensions . . 5     (0-8092)|                                     !
! SVSEA extensions . . 5     (0-8092)|                                     !
 -------------------------------------------------------------------------
 -------------------------------------------------------------------------
! Size parameters are initialized from service definition in the WLM couple  !
! data set. (IWMAM861)                                                       !
 -------------------------------------------------------------------------
```

For recovery purposes, you should define an alternate WLM CDS (similar to the sysplex alternate CDS). You can define an alternate WLM CDS using the same method (either the ISPF application or the XCF utility), but specifying a different data set name. You should also define one spare WLM CDS. See the section entitled “Increasing the Size of the WLM CDS” in *z/OS MVS Planning: Workload Management*, SA22-7602, for information about how to allocate the new CDSs.

3. WLM manages a service class period as a single entity when allocating resources to meet performance goals. You can define a *maximum* of 100 service classes in the WLM policy, and therefore there is a limit of 100 service classes in the whole sysplex.

   If your target is a PlatinumPlex, where all workloads are potentially distributed across all the systems in the sysplex, you should aim for a minimum number of service classes. If your target is a BronzePlex or a GoldPlex, where the incoming system will not share any workloads with the existing systems, it may be more effective to have a separate set of service classes for the non-system (that is, non-SYSTEM and non-SYSSTC service class) workloads in the incoming system.

   We recommend that you analyze the incoming system’s workloads and service classes and bring them into line with the target system’s service classes and classification rules where appropriate. Simply creating a superset policy, which contains all the service classes from both policies, could result in WLM being less responsive because of the large number of service classes, especially if the workloads in the sysplex run across all the systems.

   To help you merge the WLM service classes, it is helpful to map out the service classes of the different policies into a matrix, ordering the service classes by importance as shown in Table 4-2 on page 58. This matrix will help you identify service classes with duplicate names and service classes with similar goals, and gives you a good overview of how the various workloads relate to one another with regard to their respective objectives. Remember the recommendation that for maximum efficiency, there should be not more than 30 active service class periods on a system at a given time.

   In the interests of readability, we have not included the service class SYSTEM in this matrix; however, when doing your own table, you should include this service class. We *do* show the SYSSTC service class to position it in relation to the other service classes. Remember that SYSSTC has a dispatching priority of 254, which places it ahead of all importance 1 work.

   In the matrix, we also do not include additional attributes such as CPU critical or the duration of each period; however, these must be addressed during your analysis. In our example, the incoming system’s service classes are represented in bold characters; however, when doing your own matrix, it might be a good idea to use different colors to differentiate the incoming system’s service classes from those of the target sysplex.

   When comparing the service class definitions in the two environments, you need to be sure to consider the following attributes of each service class:

   – Its name. Does the same service class name exist in both environments?
   – The number of periods in the service class, and the duration of each period. You may have the same service class name in both policies; however, if the period durations are very different, the performance delivered could differ significantly.
   – The Importance for each period.
   – The target for each period.
   – Whether the “CPU critical” indicator is set.
   – Whether working set protection is used for workloads assigned to this service class.
   – The work assigned to the service class. This is discussed further in item 4 on page 59.

Table 4-2 Mapping out WLM service classes

SYSSTC		IMP1	IMP2	IMP3	IMP4	IMP5	DISC
JES2		ONLINES VEL 60		TESTONL VEL 40			
JES2		**CICSHIGH P1 90% in 0.5% sec**	**CICSLO P1 80% in 2 sec**				
LLA		EXPRESS VEL 50					
LLA		**HIGHPRTY VEL 90**					
NET			DDFTRAN P1 85% in 1 sec		DDFTRAN P2 VEL 30		DDFTRAN P3
NET				**DDFDEF P1 80% in 2.5 sec**	**DDFDEF P1 VEL 20**		
VLF			**DDFHI GH P1 90% in 1 sec**	**DDFHIGH P2 VEL 40**			
VLF			UNIXHI VEL 50		UNIXTRAN P1 VEL 30	UNIXTRAN P2 VEL 20	UNIXTRAN P3
RACF			**UNIX P1 VEL 30**	**UNIX P2 VEL 20**	**UNIX P3 VEL 10**		
RACF		TSOHI P1 R/T 90% in 1 sec		TSOHI P2 VEL 30			
TCPIP		**TSOUSER P1 90% in 1 sec**	**TSOUSER P2 VEL 70**				
TCPIP				TSOAPPL P1 R/T 85% in 2 sec	TSOAPPL P2 VEL 20		
RMF		SERVERS VEL 60	STCHI VEL 60	STCMED VEL 30	STCSLOW VEL 20		
RMF							PGN99
RMFGAT							**LIMIT**
RMFGAT			HOTBATCH VEL 40				
TRACE				**BATCHHI VEL 30**	**BATCHHI VEL 20**		
TRACE				BATCH P1 VEL 20	BATCH P2 VEL 10		BATCH P3
SYSBMAS					HOTTEST VEL 30	BATCHTST VEL 20	BATCHTST

In Table 4-2 on page 58, for each of the policies to be merged, you can begin to see where similar service class periods and workloads are defined and where you can begin the merging process. The objective is to combine into one service class those workloads you know have similar attributes and behavior. For some workloads, like TSO, this may be very obvious, but for other workloads such as batch, the task of determining the workload characteristics may require further analysis. Having RMF data available to run reports on these service classes provides you with the data necessary to make a decision on what type of service class would be best suited for the workload. You may discover that an existing service class or service class period is adequate for the incoming system workload. Or you may need to create a new one to support the new work. Customers should choose a busy time frame (between 1 and 3 hours) as a measurement period for analysis.

The RMF spreadsheet reporter can be a useful tool for this purpose. It can produce overview reports, as can the postprocessor which can be used to capture the most meaningful metrics. The best report to gather initially from an RMF perspective is the service class period report which is created using the following control card with the RMF postprocessor JCL: SYSRPTS(WLMGL(SCPER)). Information about this report format and what the reported data values mean can be found in the *z/OS Resource Measurement Facility Report Analysis,* SC33-7991. Another source of useful information is *Resource Measurement Facility Performance Management Guide,* SC33-7992.

4. Do the classification rules clash? For example, on the incoming system, do you assign CICP* to service class ONLINEP, and assign it to service class PRODONL on the target sysplex? If so, you have two choices:
   - You can merge the ONLINEP and PRODONL service classes into a single service class. However, you have to make sure that the goals for the two service classes are compatible (similar importance and target velocities/response times).
   - You can use new support delivered in OS/390 2.10 to let you assign different service classes, depending on which system the workload is running on.

   We already discussed merging service classes, so here we will just discuss assigning service classes based on which system (or group of systems) the work is running on.

   First, you have to define the classification group using the system name (**SY**) or system name group (**SYG**). After this, define the classification rule as shown in Example 4-2.

Example 4-2 Use of system name group in classification rules

```
Subsystem-Type  Xref  Notes  Options  Help
------------------------------------------------------------------------------
               Modify Rules for the Subsystem Type     Row 1 to 11 of 20
Command ===> _____________________________________________ SCROLL ===> PAGE

Subsystem Type . : STC         Fold qualifier names?   Y  (Y or N)
Description  . . . Started Tasks classifications

Action codes:   A=After     C=Copy        M=Move     I=Insert rule
                B=Before    D=Delete row  R=Repeat   IS=Insert Sub-rule
                                                       <=== More
          --------Qualifier--------           Storage    Manage Region
Action    Type       Name     Start           Critical   Using Goals Of
____  1  SYG        INCOMING ___             NO         TRANSACTION
____  2    TN         CICSPROD ___           YES        REGION
```

The use of system name or system name group is supported for address spaces whose execution system is known at classification time (ASCH, TSO, OMVS, STC). It is not supported for CICS or IMS, for example, so you cannot classify CICS or IMS transactions by system group. In Example 4-2 on page 59, we are saying that all the started tasks in the system group called INCOMING should be managed to transaction goals (where applicable), except the CICSPROD region which is to be managed to region goals.

JES is not supported because the system on which JCL conversion occurs (which is where classification occurs) may not be the system on which the job runs. Subsystem-defined transactions (CICS or IMS) and enclaves (the remainder) are not bound to an execution system at classification time either.

5. WLM provides a capability to group similar entities into Classification Groups. In the classification rules, you can then use the classification group name to assign all those entities to a given service class. This makes the classification rules much easier to maintain and understand.

   When preparing for the merge, you should review the classification groups that are defined in each environment. Are there duplicate group names? If so, do they contain entities that you will wish to assign to the same service class in the new environment? If they do, you simply need to expand the classification group in the target sysplex to include the entities being assigned to the same-named group in the incoming system.

   On the other hand, if there are groups with identical names, but you do *not* wish to assign the related entities to the same service class, you will need to set up a classification group with a new name in both the incoming system and target sysplex, and move all the entities into this new group. Remember to update the classification rules to pick up the new group name.

   If there are classification groups in the incoming system that are not in the target sysplex, you may be able to just add those groups to the target sysplex exactly as they are defined in the incoming system. Once they are defined, you should then use them in the classification rules to assign them to the appropriate service classes.

   Finally, you need to check the *contents* of the classification groups on both environments. Are there any entities that are assigned to one group in one environment, but assigned to a different group in the other environment? If so, you must resolve this conflict in advance of the merge.

6. A workload, in WLM terms, is a named collection of work to be reported as a unit. Specifically, a workload is a collection of service classes. In RMF, you can request a report for all workloads or a named workload or workloads.

   When merging WLMs, you need to check to see what workload names you have defined in each environment, and then check to see which service classes are associated with each workload. There is a good likelihood that the workload names on the different WLMs will be different. In this case, you can define the workloads names that are used in the target sysplex in the incoming system prior to the cutover. When you have defined the new workload names, you then should go through and re-assign the service classes to the appropriate workload. This will give you an opportunity to detect and resolve any problems that might arise as a result of the change. As workloads are purely a reporting entity, they have no effect on performance, and will probably only be used in performance or service level reporting jobs.

7. You need to check for duplicate report class names in both environments. Given that report classes normally have names that identify the business entity being reported on (like BILLCICS or PAYRIMS), you may not have many clashes. Regardless, you must check for duplicates and address any that you find. Any that only exist in the incoming system must be added to the target sysplex, and you have to check the classification rules

in the target sysplex to make sure that the same set of work is assigned to the report class in the target sysplex as is being assigned currently in the incoming system.

8. If the work in both the incoming system and the target sysplex will be using the same service class name, but you want to be able to continue to report separately on the workloads from each environment, update the classification rules to assign the same service class to each workload but with a different report class. This allows you to continue to report on the performance and goal achievement of each workload, but allows WLM to function more efficiently. See the following example:

SYSTEM	RULE	SERVICE CLASS	REPORT CLASS
incoming system	TN IBMU*	BATCHHI	RINBATCHH
target sysplex	TN JCO1*	BATCHHI	RTGBATCHH

   Remember that if you created a new report class in your sysplex, you may need to change your reporting/chargeback applications to get the correct values for the chargeback process.

   There were enhancements to reporting classes in z/OS 1.2 so that it now provides information at the service class period level, which was not supported previously. In addition, reporting classes can now provide response time distribution information, whereas previously, only average response times and counts were provided. Note that to get meaningful performance information reported in a report class, all the work in the report class should be of the same type, and ideally in the same service class.

9. If you currently use WLM resource groups in either environment, you must review their impact when you add another system to the sysplex. Remember that the number of service units specified in the resource group is the number that can be used *in total across the whole sysplex*. If you have a service class in the target sysplex that is assigned to a resource group, and some work in the incoming system will be assigned to the same service class, you now have more work trying to get a share of the same limited amount of resource. This will impact not only the work on the incoming system, but also any existing work in the target sysplex, as that work will now be granted less resource.

   If you decide that you want to assign more work to a service class that is associated with a resource group, check if the resource group definitions are adequate to handle the total amount of new work. If you are going to increase the limit for the resource group, you may want to hold that change until the actual time of the cutover—otherwise, work in the associated service classes will be given more resource, and hence better service, but only until the day of the cutover.

   Also, remember that resource groups can also be used to guarantee a minimum amount of service that the associated work is to receive. Once again, this value applies across the whole sysplex. So, if there will be more systems with work that is covered by that resource group, the guaranteed minimum will have less impact because that has to be shared around more work.

   For more information about resource groups, refer to *z/OS MVS Planning: Workload Management,* GA22-7602.

10. This consideration only applies if you use OPC's WLM Support capability to assign a different service class to work that is behind its target. If you use this capability, and change service class names as part of the merge, remember to update OPC with the revised service class names.

11. A scheduling environment is a list of resource names, along with their required states, that allows scheduling of work in an asymmetric sysplex. If a system image satisfies all the scheduling environment requirements associated with a unit of work, then that work can be started on that image. If any of the resource requirements are not satisfied, the unit of

work cannot be assigned to that z/OS image. Scheduling environments and resource names reside in the service definition and can be used across the entire sysplex. Each element in a scheduling environment consists of the name of a resource, and a required state, which is either ON or OFF. Each resource represents the potential availability of a resource on an individual system. The resource can represent an actual physical entity such as a database management system, or it can be an intangible quality such as a time of day. You can have up to 999 unique scheduling environments in a service definition.

If both the incoming system and the target sysplex use scheduling environments, you must check for duplicate resource names. If there are duplicate names, you must check to see if they represent the same resource. For example, if you have a resource name of CICPDOWN in both environments, you may find that a batch job that normally runs on the incoming system when CICS on that system is down might now start on a different system when the CICS on *that* system goes down - in this case, duplicate resource names would be a problem. On the other hand, if the resource name represents something like the time of day, that applies equally to all systems, then a duplicate name should not cause a problem.

If your target is a PlatinumPlex, and you are not using scheduling environments today, we recommend that you consider the use of them to make sure that jobs in the merged configuration run in the desired system(s). If your target is a BronzePlex or a GoldPlex, the JES of the incoming system is unlikely to be merged with the JES of the target sysplex, so you should not have any problems controlling where the jobs will run after the merge.

12. An application environment is a group of application functions requested by a client that executes in server address spaces. WLM can dynamically manage the number of address spaces to meet the performance goals of the work requests. Each application environment should represent a named group of server functions that require access to the same libraries. The following conditions are necessary for application environment usage:
    - The work manager subsystem must have implemented the WLM services that make use of application environments; today, this includes:
        - DB2 (subsystem type DB2) for stored procedure requests
        - SOMobjects® (subsystem type SOM®) for SOM client object class binding requests
        - WebSphere AS (subsystem CB) for Component Broker object method requests
        - Internet Connection Server (subsystem type IWEB) for Hyper-Text Transfer Protocol (HTTP) requests
    - One or more application environments must be defined in the service definition.
    - The subsystem's work requests must be associated with the appropriate application environment.

    If both systems use application environments, check the application name and subsystem names for duplicates. If the application environment is not defined in the target sysplex, you must define it before merging the systems. If only the target sysplex uses it, no changes are required.

13. WLM can dynamically manage the number of batch initiator address spaces in a JES2 or JES3 environment. You can turn over control of the batch initiator management to WLM on a job class-by-job class basis. WLM starts new initiators, as needed, to meet the performance goals of this work.

    By specifying or defaulting MODE=JES on the JES2 JOBCLASS statement or the JES3 GROUP statement, you indicate that the initiators for that job class should be JES-managed, as in the past. By specifying MODE=WLM, you put that class into WLM-managed mode. If you specify MODE=WLM for a job class, the number of initiators serving that class will be controlled by WLM on *all* the systems in that JES complex.

If you are not using WLM-managed initiators in either environment today, we recommend that you postpone implementing them until after the merge.

If you *are* using WLM-managed initiators, and your target is a BronzePlex or a GoldPlex where the JES of the incoming system will not be merged with the JES of the target sysplex, then it is possible to have WLM control the number of initiators for a given class in one JES complex but not in the other (that is, one MAS can specify MODE=JES for a class, and the other can specify MODE=WLM). Jobs submitted within each JES complex execute only on the images that are members of that complex.

If your target is a PlatinumPlex, where the incoming JES will be merged into the same JES complex as the existing systems, then the initiator control that is currently in use on the target sysplex will also apply to the incoming system as soon as it is merged.

You can limit the number of jobs in each class that can execute simultaneously in the MAS by using the XEQCOUNT=MAXIMUM= parameter on the JOBCLASS statement. This applies to JES2- and WLM-managed job classes, and can be altered with the $T JOBCLASS command. There is no facility to control the number of jobs in execution by class on an individual system basis. Similarly, there is no way to stop WLM-managed initiators on just one or a subset of systems. Of course, you could use SYSAFF= or scheduling environments to limit where the job can run, but that requires an overt action to implement.

As of OS/390 2.10 you may also have different WLM classification rules for each JES MAS using the “subsystem collection” qualifier type. For example, using this qualifier, you can assign class A jobs to a different service class in each MAS. If you need to restrict the execution of a job to a subset of images in the same MAS, then you could use scheduling environments or system affinity.

Note that WLM uses the Performance Index (PI) of a service class when deciding if it should start more initiators for a given job class. Therefore, we recommend that a given service class should not include jobs that run in both WLM-managed and JES-managed initiators. If the jobs in the JES-managed classes are the ones that are causing a poor PI, WLM may try to improve the PI by starting more initiators, which in this case may not have the desired effect.

z/OS 1.4 was announced just before this book went to press. One of the significant enhancements in that release is a change to WLM to make it far more responsive in balancing the work across multiple systems in the sysplex when using WLM-managed initiators. If you have significant amounts of batch work that runs on multiple members of the sysplex, you should investigate whether it would be wise to wait until this release is installed before starting extensive use of WLM-managed initiators.

14. Are either the incoming system or the target sysplex using response time rather than velocity goals for CICS and IMS? Prior to OS/390 2.10, once you start using response time goals for *any* CICS or IMS region, response time goals will apply to *all* CICS or IMS regions in the sysplex. In OS/390 2.10 and subsequent releases, when you classify CICS or IMS in the STC classification rules, you can specify whether the named region should be managed using TRANSACTION (response time) or REGION (velocity) goals. An example is shown in Figure 4-1 on page 64. In the example shown, the CICS region called #@$C1T2A will be managed to region goals, but all the other CICS regions in system #@$2 will be managed to transaction goals. Note that TRANSACTION is the default, however, it only applies to started tasks that support that type of goal—CICS or IMS, for example. Even though TRANSACTION is specified for the SYSTEM service class, all the started tasks in that class will actually be managed to REGION goals because they don’t support TRANSACTION goals.

If you are using response times goals for CICS or IMS in either environment, you need to decide how you want to handle things after the merge. If you do not wish to manage all your regions in this manner, you should determine the regions that you wish to manage with velocity goals and identify them as such in the classification rules. If you *do* wish to manage all your regions in this way, you should review the classification rules for the CICS and/or IMS Subsystem types and ensure that the goals specified are appropriate. In a BronzePlex or a GoldPlex, you may decide to use different service classes for the regions in the incoming system, assuming this does not cause you to exceed the maximum number of service classes in the WLM policy.

```
Subsystem-Type  Xref  Notes  Options  Help
------------------------------------------------------------------------------
                Modify Rules for the Subsystem Type     Row 1 to 11 of 20
Command ===> _____________________________________________ SCROLL ===> PAGE

Subsystem Type . : STC         Fold qualifier names?   Y  (Y or N)
Description  . . . Started Tasks classifications

Action codes:   A=After     C=Copy        M=Move     I=Insert rule
                B=Before    D=Delete row  R=Repeat   IS=Insert Sub-rule
                                                    <=== More
          --------Qualifier--------             Storage    Manage Region
Action    Type       Name     Start              Critical   Using Goals Of

 ____  1  SPM        SYSTEM   ___                NO         TRANSACTION
 ____  1  SPM        SYSSTC   ___                NO         TRANSACTION
 ____  2    TN         D#$*     ___              NO         TRANSACTION
 ____  2    TN         I#$*     ___              NO         TRANSACTION
 ____  1  TN         #@$C1T2A ___                NO         REGION
 ____  1  SY         #@$2     ___                NO         TRANSACTION
 ____  2    TN         #@$C*    ___              NO         TRANSACTION
 ____  1  TNG        SYST_TNG ___                NO         TRANSACTION
 ____  1  TN         MQ%%MSTR ___                NO         TRANSACTION
 ____  1  TN         PSM%MSTR ___                NO         TRANSACTION
 ____  1  TN         CB*      ___                NO         TRANSACTION
```

Figure 4-1 Assigning velocity goals to some CICS subsystems

15. CICSPlex® Systems Manager (CICSPlex SM)

Starting with the V1R3 release, CICSPlex Systems Manager (CICSPlex SM) is a component of CICS TS for OS/390 that monitors CICS regions and manages a CICS multiregion option (MRO) environment (prior to CICS TS for OS/390 R3, CICSPlex SM was a separate program product). CICSPlex SM provides a single image in a CICS/MRO environment through:

- Single point of control for all CICS regions
- Dynamic workload balancing
- Single system image (operations, monitoring, real time analysis)

In WLM goal mode you still have the choice of running the CICS transactions with the “join shortest queue” algorithm. This is sometimes referred to as “queue mode”. The current recommendation is to use queue mode if all transactions are achieving their goal. As the loads increase and some transactions start missing their goal, this may be a good time to move to CICSPlex SM Goal mode. While the overhead is slightly higher, this is compensated by the improved service to the really critical transactions. Note, however, that in order to use CICSPlex SM in Goal mode, you *must* be using *average* response time goals in WLM. Most customers use percentile response time goals, which are more

effective than average response time goals when there are a small number of long-running transactions in the service class. If you decide to use CICSPlex SM in Goal mode, you must bear in mind that changing WLM from percentile response time goals to average response time goals may have an adverse effect on WLM effectiveness.

If you decide that you want to implement CICSPlex SM Goal mode, the TORs must be able to communicate with WLM to get the service classes associated with their transactions. When CICSPlex SM is operated in Goal mode, the following events occur:

a. A transaction arrives at a CICS terminal-owning region (TOR).

b. The TOR passes the transactions external properties, such as LU name, userID, and so on, to the z/OS WLM.

c. z/OS WLM uses this information to assign a service class. The service class name is passed back to the TOR.

d. The TOR calls DFHCRP for transaction routing. Amongst other information, the service class name is passed in a comm_area.

e. DFHCRP in turn calls EYU9XLOP (CICSPlex SM).

f. If CICSPlex SM does not already have information about the goal for that service class, it will request that information from WLM.

g. Having the goal of the transaction, CICSPlex SM selects the “best” AOR. The name of this AOR is passed back to the TOR, which then routes the transaction to the selected AOR. These AORs could be in the same image, or in another image in the Parallel Sysplex, or even in a remote z/OS.

What are reasonable goals?

Do not expect any form of magic when you define response time goals for transactions running in a CICSPlex SM environment. CICSPlex SM and the z/OS WLM cannot compensate for poor design. If a transaction demonstrates poor response time because of poor design or coding, or simply because of the amount of processing it needs to do, there is no benefit in setting unrealistic goals with the expectation that workload management will fix the problem!

When CICSPlex SM runs in queue mode, CICSPlex SM routes the transaction to the AOR with the shortest queue independently of what is specified in the z/OS WLM. When CICSPlex SM runs in goal mode, CICSPlex SM calls the z/OS WLM to get the goals associated with a transaction. CICSPlex SM gets the response time goal, but not the importance.

CICSPlex SM does not call the z/OS WLM for every transaction, but builds a look-aside table. When a transaction is initiated, CICSPlex SM first checks to see if the goals for that transaction are already in the table, and only if CICSPlex SM does not find it there does it call the z/OS WLM. CICSPlex SM keeps track of how healthy various AORs are, and how well those AORs have done at executing transactions, to assist in deciding to which of the available AORs to route a transaction. CICSPlex SM then makes its recommendation to the TOR on where the transaction should run.

We recommend that you merge CICS service classes, using different report classes to get information on the performance of the service classes for each system. Be careful to define reasonable goals.

Table 4-3 Mapping CICS WLM goals

	Subsystem type	**CICS**		
	CICS TRANSACTION LEVEL RULES			
	QUALIFIER		CLASSES	
	TYPE	NAME	SERVICE	REPORT
1	**SI**	**CICSTA***	**CICSDEF**	**RTCCDEF**
2	TN	CEMT	CICSHIGH	RTCCEMT
2	TN	FP*	CICSLO	RTCCFPAG
1	**SI**	**CICSIN***	**CICSDEF**	**RICCDEF**
2	TN	CEMT	CICSHIGH	RICCCEMT
2	TN	FP*	CICSLO	RICCFPAG

In Table 4-3, for each rule defined we use the same service class for the transactions defined in both CICS regions (target sysplex and incoming system), but we associated different report classes with each one for future analysis.

16. The Intelligent Resource Director.

Intelligent Resource Director (IRD) extends the concept of goal-oriented resource management by allowing you to group z/OS system images that are resident on the same CPC running in LPAR mode, and in the same Parallel Sysplex, into an “LPAR cluster.” This gives WLM the ability to manage processor and channel subsystem resources, not just in one single image but across all the images in the LPAR Cluster.

IRD is described in detail in the IBM Redbook *z/OS Intelligent Resource Director,* SG24-5952. However, there are a number of salient points that we want to reiterate here:

- A z/OS system *automatically* becomes part of an LPAR cluster when it is IPLed on a zSeries processor. The LPAR cluster name is the same as the sysplex name. If more than one system in the same sysplex is IPLed on a given CPC, all those systems that are in the same sysplex will automatically be part of the same LPAR cluster. There is nothing you have to do to make this happen, and equally, there is nothing you can do to *stop* this happening.
- The WLM LPAR CPU Management feature can be turned on and off on an LPAR by LPAR basis using the HMC.
- The Channel Subsystem I/O Priority queueing feature is turned on or off at the CPC level. Once it is turned on, the I/Os from *all* LPARs will be prioritized in the channel subsystem.
- Dynamic Channel-path Management (DCM) is automatically enabled if there are managed channels and managed control units in the configuration of the LPAR. You cannot decide to use DCM in one LPAR in an LPAR cluster, but not in another LPAR in the same cluster.

Bearing these facts in mind, you should consider the following:

- If the incoming system is on the same CPC as some of the systems in the target sysplex, you now have the *option* (it is not mandatory) to use WLM LPAR CPU Management in that LPAR.

- If you decide to use WLM LPAR CPU Management for a given LPAR, you must make sure that your WLM goals would make sense if *all* the work in the LPAR Cluster were running in one z/OS image—this is not the same as merging the WLM policies. You can have a merged policy that works fine as long as Development and Production never run in the same image, however, WLM LPAR CPU Management effectively extends the idea of an image to cover all the images in the LPAR Cluster. You must make sure that the relative Importance and goals of all the work in the LPAR Cluster make sense in relation to each other.
- If you are currently using DCM in the LPAR Cluster, remember that there will now be another system that is able to use the managed paths. Therefore, you may want to review the number of managed channels, bearing in mind the load and the workload profile of the incoming system.
- With DCM, managed channels can only be used by LPARs that are in the same LPAR Cluster. If the control unit is shared with LPARs outside the LPAR Cluster, you need to keep a number of non-managed channels that those LPARs can still use. By moving the LPAR into the sysplex, you *may* reduce the number of non-managed channels that you need for the control unit.

  Alternately, if you have a control unit that is a candidate for DCM, but you have not been using DCM because of the need to share the control unit with LPARs outside the LPAR Cluster, moving the incoming system into the sysplex (and therefore into the LPAR Cluster) may allow you to start using DCM with that control unit.

Before and after making the changes, extract RMF reports for both systems (incoming system and target sysplex). Verify the goals and redefine them, if necessary. It is very important to maintain the goals with the same availability and performance.

17. Parallel Access Volumes (PAV) is a capability delivered with the IBM ESS (2105) to let you have more than one UCB for a given device. You can now have what is known as a “base UCB”, which is always associated with the device, and “alias UCBs”, which can potentially be moved between base devices. Having more than one UCB gives you the ability to have more than one I/O going to the device at one time.

    OS/390 2.7 introduced the ability for WLM to dynamically manage the movement of aliases among devices, based on the IOS queueing for each device, and the importance of the work on each device. This feature is known as dynamic PAV or dynamic alias management.

    WLM uses information about all the systems in the sysplex that are using a device when deciding what to do about an alias; however it has no knowledge of how systems outside the sysplex are using the device. For this reason, you should not use dynamic PAV with a device that is shared outside the sysplex.

    If merging the incoming system into the sysplex means that all sharing systems are now in the same sysplex, you might be able to enable dynamic PAV for that device now.

18. Before you start work on merging the systems, you should review RMF Workload Activity reports for the incoming system and all the existing systems in the target sysplex. You should especially review the system PI for each service class that is active on each system. Just because there are workloads on three systems that are using the same WLM service class does not mean that each system is *achievin*g the goal. If you find a service class that rarely achieves its goal on a given system, you should review whether the workload assigned to that service class should perhaps be assigned to another, more realistic one. This will help you ensure, prior to the merge, that the work in each service class is appropriate that service class, and that the goals are actually being achieved.

    Following the merge, the RMF reports should be checked again to ensure that all the goals are still being met.

19. OS/390 Release 10 addresses user requirements to set performance goals based on the run-time location of the work. This release provides the ability to classify work based on system name (or named groups of them).

    Classification rules are the rules you define to categorize work into service classes, and optionally report classes, based on work qualifiers. A work qualifier is what identifies a work request to the system. The first qualifier is the subsystem type that receives the work request.

    There is one set of classification rules in the service definition for a sysplex. They are the same regardless of what service policy is in effect; a policy cannot override classification rules. You should define classification rules after you have defined service classes, and ensure that every service class has a corresponding rule. Remember that to put the new classification rules into effect, you need to install the service definition into the CDS and then activate the policy.

    The list of work qualifiers introduced in Release 10 and their abbreviations are:

    SY System name
    SYG System name group
    PX Sysplex name
    SSC Subsystem collection name
    SE Scheduling environment name

    Note: Different subsystem types support each qualifier type.

    – The PX qualifier is supported by all IBM-defined subsystem types.
    – The SE is supported by DB2 and JES.
    – The SSC is supported by DB2, DDF and JES.
    – The SY and SYG are supported by ASCH, OMVS, STC and TSO.

    If you decide to use any of the new qualifiers, remember that the order of the classification rules is critical. If you wish to have more specific assignments, such as those that pertain to just a subset of systems, you must place those assignments first in the rule statements.

20. It is not uncommon for WLM constructs (service class, workload, report class) to be used in service reporting and chargeback reports. You should review your chargeback and accounting jobs to see what information they use to associate a given piece of work with a reporting entity. If any of the WLM constructs are used, you need to make sure that any changes in the WLM constructs assigned to a given piece of work are reflected back into the reporting jobs. If you follow our recommendations and make the changes in the WLM policy of the incoming system before the actual cutover, then there should be no surprises following the cutover, when all systems are using the same WLM policy.

4.4 Methodology for merging WLMplexes

When merging two WLMplexes, all of the merge work is really done *before* the actual merge of the incoming system into the target sysplex takes place. If you investigate and address all of the considerations listed in 4.3, “Considerations for merging WLMplexes” on page 54, the actual merge should be uneventful.

There are basically two approaches you can take. The first is to simply take all the service classes that are defined in the WLM for the incoming system and add those to the WLM service policy that is currently in use in the target sysplex. However, such an approach is unlikely to be successful for the following reasons:

- It is likely that there will be service classes with the same names in both policies, and those classes may be defined differently.

- You will have to decide how to handle default service classes - do you use the one from the incoming system, or the one currently used in the target sysplex?
- The classification rules for the two policies can, and probably will, clash. That is, on the incoming system you may assign JES job class A to PRODHOT, while on the target sysplex it may be assigned to TESTSLO.
- You will definitely have SYSTEM and SYSSTC service classes in both policies, but the address spaces assigned to each may be different across the two policies.

So, while it may initially appear easier just to create a superset of the two policies, by the time you are finished, it will have taken as much effort as if you had merged the service classes.

Which brings us to the other approach: merge the service classes from the two policies. This may seem like more work up front, but it will result in a system that provides better performance and is more manageable.

If you work through the checklist in 4.3, “Considerations for merging WLMplexes” on page 54, by the time of the merge, there should be no further work required. The one thing you must do *immediately before and after* the merge is to check the RMF Workload Activity reports for each service class in each system in the plex. The objective is to ensure that bringing the system into the sysplex has not had any adverse impact on performance.

4.5 Tools and documentation

In this section we provide information about tools and documentation that may help you complete this task.

4.5.1 Tools

The following tools are available, and may help you prepare for the merge of the WLMs:

- The WSC Migration Checklist is available at:

  http://www.ibm.com/servers/eserver/zseries/zos/wlm/pdf/wsc/pdf/v2.0_guide.pdf
- The WLMMIGR tool is available at:

  http://www.ibm.com/servers/eserver/zseries/zos/wlm/migration/migration.html
- The tool to print WLM policy in HTML format is available at:

  http://www.ibm.com/servers/eserver/zseries/zos/wlm/faqs/faq00002.html#faqb9

4.5.2 Documentation

The following documentation may be useful during the merge process:

- *Resource Measurement Facility Performance Management Guide,* SC33-7992
- *z/OS Intelligent Resource Director,* SG24-5952
- *z/OS MVS Planning: Workload Management,* GA22-7602

5

GRS considerations

This chapter discusses the following aspects of moving a system that is using GRS to manage multi-system resource serialization into a sysplex where all the existing systems are using GRS:

- Is it necessary to merge GRS environments, or is it possible to have more than one GRS environment in a single sysplex?
- A checklist of things to consider should you decide to merge the GRSs.
- A methodology for doing the merge.
- A list of tools and documentation that can assist with this exercise.

5.1 One or more GRSplexes

A GRSplex is a collection of z/OS systems that use GRS to serialize access to shared system resources such as data sets on shared DASD volumes. All the systems in the GRSplex must execute in the same GRS mode, either Ring or Star.

It is *not* possible to have more than one GRSplex in a single sysplex. Therefore, you *must* merge the incoming system into the existing GRSplex.

We recommend that you execute in GRS Star mode. In a Parallel Sysplex, Star mode offers significant performance and availability improvements over Ring mode.

5.1.1 Star complex

If you are executing in GRS Star mode, you *must* merge the incoming system into the existing GRSplex because the systems in the Star GRSplex must all be in the same Parallel Sysplex.

5.1.2 Ring complex

There are two types of GRS Ring configuration:

Sysplex matches complex

In this configuration, you are executing in Ring mode, and all the systems in the GRSplex are also in the sysplex. If you are in this configuration, you may wish to convert your GRSplex to Star mode prior to merging your incoming system into the sysplex. GRS Star provides better performance than GRS Ring in nearly every case, with the difference in performance increasing as the number and size of the systems in the sysplex grows.

Mixed complex

Another possibility, if you are executing in Ring mode, is that the GRSplex contains systems that are not in the sysplex. This configuration is called a mixed GRS complex. In this environment, those systems that are in the sysplex will use XCF for GRS communication, while those systems that are outside the sysplex will need their own dedicated CTCs between each other and each of the systems in the sysplex to participate in the GRSplex.

5.1.3 Target environments

Regardless of whether your target environment is a BronzePlex, a GoldPlex, or a PlatinumPlex, you must have just one GRSplex for all the systems. There may be some considerations for specific products if you do not merge them (HSM is an example), however, those will be reflected in the Resource Name Lists (RNLs). For GRS itself, there is only one option—to include all the systems in a single GRSplex.

There is an additional consideration for GRS in a BronzePlex or a GoldPlex related to duplicate data set names. Because everything is shared in a PlatinumPlex, you are less likely to have duplicate data set names in that environment. And because the user DASD are not shared in a BronzePlex or a GoldPlex, it may be felt that duplicate data set names are less of a problem in those environments. However, the presence of duplicate data set names introduces the possibility of getting false contention in GRS, where two different data sets that have the same name will appear to GRS as being the same resource, and therefore it will only allow one of them to be updated at a time. Some customers have looked at using the

GRS ISGNQXIT exit to change the resource name based on which subplex the resource resides in; however, there are problems with this approach in that GQSCAN does not see the modified resource name, and the **D GRS** command also does not show the modified resource name.

5.1.4 Why convert to a Star complex

A Star complex offers many improvements over a Ring complex, such as:

- Improved global request response time
- Reduced real storage usage
- Reduced CPU consumption
- Improved availability and recovery

In addition, a Star complex positions you for future expansion of the sysplex.

A Star complex is built around a CF lock structure. GRS uses a lock structure called ISGLOCK in the CF to store information about all the ENQ and DEQ requests for global resources in the GRSplex. Each system maintains a local copy of only its own global requests. When a global request is issued, GRS uses information in the ISGLOCK structure to determine and assign ownership of a particular resource across all the systems in the GRSplex.

In contrast, the Ring consists of one or more systems connected to each other by communication links. The global resource requests are passed from one system to another in a message or token called the ring system authority message (RSA). There is one RSA token in a Ring complex and it is passed from system to system in a round-robin fashion. When a system receives the RSA, it inserts its own global ENQ and DEQ information, and passes it along to the next system to copy. It also makes a copy of the global ENQ and DEQ information that was inserted by the other systems. When the RSA returns to a system, it knows that all the other systems have seen the requests that were originated from that system, so these requests are removed and processed.

Improved response times

There are requests for two types of resources: local and global.

- When a request is made for a local resource, the ownership of the resource is serialized within the system that made the request. This request is not seen by other systems in the GRSplex. Local resources are usually requested with a scope of STEP or SYSTEM.
- When a request is made for a global resource, ownership of the resource is serialized across all the systems in the GRSplex. Global resources are requested with a scope of SYSTEMS.

As you add more systems to a GRS Ring, you increase the possibility of performance-related problems with requests for global resources because:

- Each global request takes longer to be serviced, due to the additional time taken by the RSA to circulate around the larger ring.
- Global requests might be deferred because the RSA is full.
- The number of requests that can be processed per second decreases as the ring size increases due to the increased path taken by the RSA to circulate around the ring. Each system holds the RSA for a minimum of 1 *milli*second. So the response time for a GRS request in a ring configuration is *at least* 1 millisecond times the number of systems in the sysplex.

In a Star complex, each system communicates directly with a CF instead of using the RSA to communicate with each other. A global request can be satisfied much faster, requiring as little as two signals to gain access to a resource. Typical GRS lock structure response times are in the range of 10-30 *micro*seconds, depending on the speed of the processors and links.

Reduced storage requirements

In a Ring complex, each system stores all the global requests for the entire sysplex. The amount of real storage used by GRS on each system increases with every system added to the sysplex. In a Star complex, each system stores only its own requests, and the amount of storage used by GRS is largely independent of the number of systems in the sysplex. For sysplexes containing more than two systems, Star mode generally requires less storage than Ring mode.

Reduced CPU consumption

In a Ring complex, each system maintains a view of all the global requests for the entire sysplex, in effect, processing the global requests for every system in the sysplex. In a Star complex, each system processes only its own requests, resulting in reduced CPU consumption when in GRS Star mode.

Improved availability

In a Ring complex, the ring is disrupted whenever a system fails unexpectedly. During ring rebuild, global requests cannot be processed. Additionally, as the Ring complex grows, the time to rebuild the ring increases for two reasons:

- The total number of resources managed in the ring is the sum of all the resources on each system, and each resource needs to be re-synchronized before a system can rejoin the ring.
- The number of systems that need to interact to rejoin the ring is increased. The ring is rebuilt one system at a time.

In a Star complex, if a system fails, the surviving systems do not need to communicate to continue global request processing. The failure of the system is perceived as a large DEQ request (i.e. the DEQ of all resources held by requestors on the failed system).

Requirements

To ensure availability, you should have at least two CFs that are connected to all systems in the sysplex, and have all those CFs specified on the PREFLIST for the lock structure. If the CF containing the lock structure fails, or one of the systems loses connectivity to that CF, the lock structure will automatically rebuild to an alternate CF. If any system is unable to access the lock structure, GRS will put that system into a wait state. The two CFs do not have to be failure-isolated.

If you are executing a mixed GRS complex, you will not be able to convert to Star mode. If you want to convert to Star mode, you will have to either convert the non-sysplex systems to sysplex systems, or remove them from the GRSplex. To decide which way to go, first you will have to determine the level of data sharing between the sysplex and non-sysplex systems. If there is a need to share the data, then you will have to convert the non-sysplex systems to sysplex systems. If not, then you can remove the non-sysplex systems from the GRSplex.

5.2 Considerations for merging GRSplexes

This section contains, in checklist format, a list of things that must be considered should you decide to merge two or more GRS complexes.

Table 5-1 Considerations for merging GRSplexes

Consideration	Note	Type	Done?
If the target sysplex is using GRS Ring mode, consider converting to Star mode prior to merging the incoming system into the GRSplex.	**1**	**B,G,P**	
If you are converting to GRS Star mode, ensure the sysplex CDSs in the target sysplex have been formatted with the GRS parameter.	**2**	**B,G,P**	
If you are converting to GRS Star mode, the IEASYS member of Parmlib must be updated appropriately.	**3**	**B,G,P**	
If you are already using GRS Star in the target sysplex, ensure the ISGLOCK structure is appropriately sized for the new number of systems in the sysplex.	**4**	**B,G,P**	
Review the GRS RNL of both the target sysplex and the incoming system to ensure they adhere to the current IBM and ISV GRS RNL recommendations.	**5**	**B,G,P**	
Ensure all data set names on the incoming system conform to the established sysplex-wide data set naming standards used in the target sysplex.	**6**	**B,G,P**	
Review your MPF lists, MPF exits and Automation scripts to ensure the GRS system messages are treated the same way in both the incoming system and the target sysplex.	**7**	**B,G,P**	
Apply the appropriate GRS compatibility maintenance to the incoming system and/or the systems in the target sysplex.	**8**	**B,G,P**	
All the systems in the GRSplex must use common GRS exits. Ensure the GRS exits of the incoming system are exactly the same as those in the target sysplex.	**8,9**	**B,G,P**	
All the systems in the GRSplex must use a common RNL. Ensure the RNL of the incoming system is exactly the same as that in the target sysplex. Consider using the RNL wildcard character support to make the RNL merge easier.	**9**	**B,G,P**	
If you are executing in a mixed GRS complex, set up dedicated communications links to those systems outside the sysplex.	**10**	**B,G,P**	
Provide connectivity to the target sysplex CFs and set up the appropriate Parmlib members to participate in the target sysplex.		**B,G,P**	
Update your Operator instructions and make your operators aware of changes to operational procedures.	**11**	**B,G,P**	
Review the RNLs after the cutover and remove or modify as appropriate.	**12**	**B,G,P**	

The “Type” specified in Table 5-1 relates to the sysplex target environment—**B** represents a BronzePlex, **G** represents a GoldPlex, and **P** represents a PlatinumPlex.

Notes indicated in Table 5-1 are described below:

1. We recommend converting to Star mode to prevent any potential performance problems caused by increasing the number of systems in the GRS Ring. For instructions on how to convert to a Star complex, see Chapter 6, “Steps in converting to a Star Complex” in *z/OS MVS Planning: Global Resource Serialization,* SA22-7600.

2. If you wish to use GRS in Star mode, the sysplex CDSs must be formatted (using the IXCL1DSU program) with the GRS keyword. To find out if the sysplex CDSs have already been formatted with this keyword, issue a **`D XCF,C,TYPE=SYSPLEX`** command. The response will tell you whether GRS Star mode is supported by each CDS.
3. To convert to a Star complex, you will need to change the following members in Parmlib:
   - IEASYSxx. You will need to modify the GRS parameter to GRS=STAR.
   - GRSCNFxx. If you are using the default component trace Parmlib member called CTIGRS00, this member is no longer required.

   For instructions on what members to change in Parmlib to implement a Star complex, see Chapter 5, “Defining Parmlib Members for a Star Complex” in *z/OS MVS Planning: Global Resource Serialization,* SA22-7600.
4. For instructions on how to use the GRS tool ISGSCGRS to help size the ISGLOCK structure, see Chapter 5, “Sizing the ISGLOCK Structure” in *z/OS MVS Planning: Global Resource Serialization,* SA22-7600.

   IBM provides a Web-based wizard to help you to size the ISGLOCK structure. The wizard can be found at this IBM site:

   `http://www.ibm.com/servers/eserver/zseries/cfsizer/`
5. Review the GRS RNL of both your target sysplex and your incoming system to ensure they adhere to the current IBM and ISV RNL recommendations. Your review will help you identify any redundant RNL entries which you can delete prior to merging the RNLs. You may also identify some recommendations that are missing from your RNL.

   IBM provides a list of default RNLs. This list is not very comprehensive because the resources required in each RNL tend to vary from site to site. For the list of default RNLs, see Chapter 2, “RNL Considerations” in *z/OS MVS Planning: Global Resource Serialization,* SA22-7600.

   IBM provides general recommendations, as well as specific suggestions, on known resources that are good RNL candidates. See Chapter 2, “RNL Considerations” and “RNL Candidates” in *z/OS MVS Planning: Global Resource Serialization,* SA22-7600. The list covers the following topics:

   - CICS
   - CVOLs (hopefully you are not still using these!)
   - DAE
   - DB2
   - DFSMShsm
   - DFSMS/MVS
   - IMS
   - ISPF or ISPF/PDF
   - JES2
   - JES3
   - System Logger
   - RACF
   - Temporary data sets
   - TSO/E
   - VIO journaling data sets
   - VSAM

   You will also find specific recommendations in the following manuals:

   - Chapter 2, “Preventing Lockouts on Shared Volumes” in *z/OS DFSMS:Managing Catalogs,* SC26-7409

- Chapter 9, “Serialization” in *z/OS DFSMSdss Storage Administration Guide,* SC35-0423
- The sections entitled “Converting from Volume Reserves to Global Resource Serialization” and “VSAM SHAREOPTIONS Parameters for Control Data Sets” in Chapter 11 in *z/OS DFSMShsm Implementation and Customization Guide,* SC35-0418
- Chapter 2, “Updating GRSRNLxx (Optional)” in *z/OS DFSMSrmm Implementation and Customization Guide,* SC26-7405
- Chapter 1, “Accessing JES2 Checkpoint Data Sets in a MAS” in *z/OS JES2 Initialization and Tuning Guide,* SA22-7532
- Chapter 1, “RACGLIST considerations” in *z/OS Security Server RACF System Programmer’s Guide,* SA22-7681
- Chapter 4, “Using the global resource serialization function” in *z/OS Security Server RACF System Programmer’s Guide,* SA22-7681
- Chapter 4, “Sharing a database” in *z/OS Security Server RACF System Programmer’s Guide,* SA22-7681
- Chapter 3, “Considerations for all CDSs” in *z/OS MVS Setting Up a Sysplex,* SA22-7625
- Chapter 9, “Add the DASD Data Sets to the GRSRNL Inclusion List” in *z/OS MVS Setting Up a Sysplex,* SA22-7625
- Appendix G, “Updating Parmlib for Automatic Tape Switching” in *z/OS MVS Setting Up a Sysplex,* SA22-7625
- Chapter 1, “TSO Generic Resource Support” in *z/OS TSO/E General Information,* SA22-7784

If you make any changes to your RNLs, use the Symbolic Parmlib Parser to check that your changes are syntactically correct. For more information, see Appendix B, “Symbolic Parmlib Parser” in *z/OS MVS Initialization and Tunning Reference,* SA22-7592.

6. You need to ensure that all the data set names in your incoming system conform to the established sysplex-wide data set naming standards used in your target sysplex. In particular, this applies to all the data sets that will reside on shared DASD.

   You need to use unique names for your data sets; otherwise, contention could occur on resources that happen to share the same name but are physically different.

   For more information, see Chapter 2, “Data Set Naming Conventions” in *z/OS MVS Planning: Global Resource Serialization,* SA22-7600.

7. Check your MPF lists, MPF exits, and Automation scripts to ensure the GRS system messages are treated the same way on both the incoming system and the target sysplex. The GRS system messages start with the message prefix ISG.

   If they are treated differently, determine why and make any changes that are necessary to ensure that the GRS messages are treated the same way in both environments. For more information about MPF lists see Chapter 61, “MPFLSTxx (Message Processing Facility List)” in *z/OS MVS Initialization and Tunning Reference,* SA22-7592. For more information about the MPF exits, see Chapter 10, “IEAVMXIT — Installation-Specified MPF Exits” in *z/OS MVS Installation Exits,* SA22-7593.

8. Apply the required GRS compatibility maintenance to the appropriate incoming system and/or the systems in the target sysplex.

   OW49779 is the compatibility APAR for z/OS 1.2. This APAR provides compatibility support for GRS wildcard characters in OS/390 2.8 to OS/390 2.10. Apply this maintenance to the appropriate systems if one or more of the merging systems is at z/OS 1.2 or higher.

After this APAR is installed, the RNL exit point ISGGREX0, with the entry points of ISGGSEEX, ISGGSIEX and ISGGCREX, will no longer be supported. These modules receive control whenever an ENQ, DEQ, or RESERVE request is issued for a resource. The default exits simply return to GRS without modifying the data.

During GRS initialization, if one or more of these modules are detected, message ISG351I is issued to indicate these modules are no longer supported. This message is purely informational. There is no action required if you are executing the default RNL modules.

The function provided by these modules has been replaced by a call to ISGNQXIT. If you have modified these modules, you will need to review whether you need to duplicate the function previously provided by the superseded RNL modules. For more information about ISGNQXIT, see Chapter 32, “ISGNQXIT - ENQ / DEQ Installation Exit” in *z/OS MVS Installation Exits,* SA22-7593.

A sample of the exit ISGNQXIT and its supporting documentation can be found on the IBM Redbooks site:

`ftp://www.redbooks.ibm.com/redbooks/SG246235/`

9. GRS *requires* RNL consistency across the sysplex. If a system is IPLed with RNLs that do not match the rest of the sysplex, GRS will put that system into a wait state. However the GRS exits may make different decisions on different systems—GRS does not check that the same exits are used on all systems.

   GRS processing for ENQ, DEQ, or RESERVE requests should yield the same result on every system in the GRSplex. If they don’t, resource integrity cannot be guaranteed. To ensure resource integrity, your ENQ/DEQ exit routines and your RNL should be identical on all systems in the GRSplex.

   The process for aligning the ENQ/DEQ modules and the RNL would be:

   a. If required, merge the ENQ/DEQ exit requirements into a single ENQ/DEQ exit.
   b. If required, implement the single ENQ/DEQ exit on all the merging systems.
   c. Create a single RNL, by consolidating the RNL from the incoming system and the active RNL from the target sysplex.
   d. Activate the consolidated RNL on all the merging systems.

   If you have customized the ENQ/DEQ modules, you will need to merge your requirements into a single exit. Then you will have to install this exit on the incoming system and all the systems in the target sysplex. The ENQ/DEQ exit points are:

   – ISGGREX0 (if your systems do not have APAR OW49779 installed)
   – ISGNQXIT (if your systems do have APAR OW49779 installed).

   As previously mentioned, OW49779 is the compatibility APAR for z/OS 1.2. ISGNQXIT has already been discussed in a previous consideration that discussed compatibility maintenance. For information about ISGGREX0, see Section 2, “ISGGREX0” in *OS/390 MVS Installation Exits,* SC28-1753.

   Once you have installed the exits, create a single RNL by consolidating the RNL from the incoming system with the RNL from the target sysplex.

   During GRS initialization, if there is a mismatch between the RNL on the incoming system and the active RNL in the target sysplex, the incoming system will not be allowed to join the target sysplex. The incoming system will go into a non-restartable wait state x’0A3’.

   If wildcard character support is available on all the merging systems, we recommend you use this feature, as it may make RNL consolidation easier.

Wildcard character support has another advantage. It tends to make RNLs smaller and simpler. The RNL has a maximum size of 61 k. If the merged RNL is larger than this, you won't be able to implement it. Most sites won't reach the limit. For information on determining the size of your RNL, see Chapter 16, "ISG251I" in *z/OS MVS System Messages Volume 9 (IGF-IWM),* SA22-7639.

After you have made the changes to create a single RNL that can be used on all the merging systems, use the Symbolic Parmlib Parser to check your changes are syntactically correct. For more information, see Appendix B, "Symbolic Parmlib Parser" in *z/OS MVS Initialization and Tunning Reference,* SA22-7592.

To change the active RNLs when executing in Star mode:

– Use the **SET GRSRNL** command. For more information, see Chapter 6, "Changing RNLs for a Star Complex" in *z/OS MVS Planning: Global Resource Serialization,* SA22-7600.

To change the active RNLs when executing in Ring mode, where the sysplex equals the complex:

– Use the **SET GRSRNL** command. For more information, see Chapter 9, "Changing RNLs for a Ring" in *z/OS MVS Planning: Global Resource Serialization,* SA22-7600.

To change the active RNLs when executing in Ring mode, in a mixed GRS complex:

– Remove the non-sysplex systems from the GRSplex.

– Use the **SET GRSRNL** command. For more information, see Chapter 9, "Changing RNLs for a Ring" in *z/OS MVS Planning: Global Resource Serialization,* SA22-7600.

– Re-IPL the non-sysplex systems with the new consolidated RNL.

At this point, both your incoming system and your target sysplex will be using an identical ENQ/DEQ exit and RNL.

10. In a mixed GRS complex, those systems participating in the sysplex will use XCF for GRS communication. Those systems outside the sysplex will need their own dedicated CTCs between each other and the systems in the sysplex in order to participate in the GRSplex.

    You will need to provide dedicated communication links for GRS traffic between the incoming system and the non-sysplex systems in the GRSplex. For information about configuring the link, see Chapter 8, "Designing a Mixed Complex" in *z/OS MVS Planning: Global Resource Serialization,* SA22-7600.

    When the communication links are implemented, use the following process to enable the non-sysplex systems for communication with the incoming system:

    a. Modify GRSCNFxx on the non-sysplex systems. Add the CTC parameter to specify the device numbers of the communications links to be used between the incoming system and the non-sysplex systems.

    b. IPL the non-sysplex systems so that the CTC links are enabled on those systems prior to the incoming system merging with the target sysplex.

11. You will need to update your Operator instructions and make your Operations staff aware of the following changes:

    – The output from a **D GRS** command will show *all* the systems in the merged sysplex, not only the incoming system or the systems in the target sysplex.

    – When the incoming system joins the target sysplex, the **V GRS** command will no longer be supported. To remove the incoming system from the merged sysplex, they will have to use the **V XCF,sysname,OFFLINE** command.

    For information about these commands, see Chapter 4, "Controlling a Global Resource Serialization Complex" and Chapter 4, "Removing a System from the XCF Sysplex" in *z/OS MVS System Commands,* SA22-7627.

12. If you were sharing resources between the incoming system and the target sysplex prior to the merge, you may have been using Reserves to ensure data integrity for those resources. An example is a catalog that is shared between systems that are not all in the same GRSplex. Such resources should have an entry similar to the following:

    ```
    RNLDEF RNL(EXCL) TYPE(SPECIFIC) QNAME(SYSIGGV2) RNAME(ucat.fred)
    ```

    If the incoming system is the only system outside the target sysplex that is sharing a given resource, you should be able to remove that entry from the Exclusion list *after* the merge is complete. This will remove any unnecessary Reserves.

5.3 Methodology for merging GRSplexes

The following methodology will help you to merge your incoming system into your existing GRSplex.

We recommend that you are executing in Star mode prior to merging your incoming system into your target sysplex. This is because Star mode offers significant performance improvements over Ring mode in large sysplexes. In small sysplexes, Star mode may not deliver as noticeable performance improvements; however, you are then well positioned for any future expansion of the sysplex.

5.3.1 Merging into a Star complex

Your incoming system will join the GRSplex when you IPL your incoming system into the target sysplex. After you have completed the relevant pre-merge tasks described in 5.2, “Considerations for merging GRSplexes” on page 74, use the following process to accomplish the merge:

1. Modify Parmlib.

   You will need to modify the following members in Parmlib:

   – IEASYSxx:
     - You will need to change the GRS parameter to GRS=STAR.
   – GRSCNFxx:
     - If you are using the default component trace Parmlib member called CTIGRS00, this member is no longer required.

   For the instructions on what members to change in Parmlib to implement a Star complex, see Chapter 5, “Defining Parmlib Members for a Star Complex” in *z/OS MVS Planning: Global Resource Serialization,* SA22-7600.
2. IPL your incoming system into your target sysplex.

5.3.2 Merging into a Ring complex (sysplex matches complex)

Your incoming system will join the GRSplex when you IPL your incoming system into the target sysplex. After you have completed the relevant pre-merge tasks described in 5.2, “Considerations for merging GRSplexes” on page 74, use the following process to accomplish the merge:

1. Modify Parmlib.

   You will need to modify the following members in Parmlib:

   – IEASYSxx:
     - You will need to change the GRS parameter to GRS=TRYJOIN.

– GRSCNFxx:
  - Replace this member on the incoming system with a copy of this member from the target sysplex. This is to ensure parameters such as TOLINT are consistent across the entire sysplex.

For information about what members to change in Parmlib, and the meaning of the different parameters, see Chapter 8, “Processing Options in a Sysplex” in *z/OS MVS Planning: Global Resource Serialization,* SA22-7600.

2. IPL your incoming system into your target sysplex.

5.3.3 Merging into a Ring complex (mixed GRS complex)

Your incoming system will join the GRSplex when you IPL your incoming system into the target sysplex. After you have completed the relevant pre-merge tasks described in 5.2, “Considerations for merging GRSplexes” on page 74, use the following process to accomplish the merge:

1. Modify Parmlib.

   You will need to modify the following members in Parmlib:

   – IEASYSxx:
     - You will need to change the GRS parameter to GRS=TRYJOIN.
   – GRSCNFxx:
     - Replace this member on the incoming system with a copy of this member from the target sysplex. This is to ensure parameters such as TOLINT are consistent across the entire sysplex.
     - Modify the CTC parameters to specify the device numbers of the communications links to be used between the incoming system and the non-sysplex systems.

   For information about what members to change in Parmlib, and the meaning of the different parameters, see Chapter 8, “Processing Options in a Mixed Complex” in *z/OS MVS Planning: Global Resource Serialization,* SA22-7600.

2. IPL your incoming system into your target sysplex.

5.4 Tools and documentation

In this section we provide information about tools and documentation that may help you complete this task.

5.4.1 Tools

The following tools may be of assistance to you during the merge process:

Symbolic Parmlib Parser

This tool will enable you to check that your RNL is syntactically correct. For more information, see Appendix B, “Symbolic Parmlib Parser” in *z/OS MVS Initialization and Tunning Reference,* SA22-7592.

ISGSCGRS

This tool will enable you to size the ISGLOCK structure. For more information, see Chapter 5, “Sizing the ISGLOCK Structure” in *z/OS MVS Planning: Global Resource Serialization,* SA22-7600.

Coupling Facility Structure Sizer

This Web-based wizard will help you to size the ISGLOCK structure. The wizard can be found at this IBM site:

http://www.ibm.com/servers/eserver/zseries/cfsizer/

5.4.2 Documentation

The following publications may be of assistance to you during the merge process:

- *OS/390 MVS Installation Exits,* SC28-1753
- *z/OS DFSMShsm Implementation and Customization Guide,* SC35-0418
- *z/OS DFSMSrmm Implementation and Customization Guide,* SC26-7405
- *z/OS JES2 Initialization and Tuning Guide,* SA22-7532
- *z/OS MVS Initialization and Tunning Reference,* SA22-7592
- *z/OS MVS Installation Exits,* SA22-7593
- *z/OS MVS Planning: Global Resource Serialization,* SA22-7600
- *z/OS MVS Setting Up a Sysplex,* SA22-7625
- *z/OS MVS System Commands,* SA22-7627
- *z/OS MVS System Messages Volume 9 (IGF-IWM),* SA22-7639
- *z/OS Security Server RACF System Programmer's Guide,* SA22-7681
- *z/OS TSO/E General Information,* SA22-7784

6

SMF considerations

This chapter discusses the considerations for managing SMF data for a system that is being moved into a sysplex.

6.1 Managing your SMF data

System Management Facilities (SMF) collects and records system and job-related information that your installation may use in:

- Billing users
- Reporting reliability
- Analyzing the configuration
- Scheduling jobs
- Summarizing direct access volume activity
- Evaluating data set activity
- Profiling system resource use
- Maintaining system security

SMF formats the information that it gathers into system-related 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.

It is not possible to have one system collecting all SMF data for all systems within a sysplex. Each system within the target sysplex will still need to collect system-specific SMF data. However, some SMF records are sysplex-wide and only need to be collected by one system within the sysplex. Post-processing of SMF data may change on the incoming system, depending on the configuration within the target sysplex. To help understand the SMF considerations when merging a system into a sysplex refer to Table 6-1 on page 84.

In line with the sysplex target environment options outlined in 1.2, “Starting and ending points” on page 2, moving the incoming system into a BronzePlex would typically be done to obtain the benefits of PSLC or WLC charging. This would mean that the incoming system would have limited sharing and the collection, switching, management and post-processing of SMF data would not have to change.

Moving the incoming system into a GoldPlex would be done to share the same system software environment (sysres) and maintenance environment, and therefore you will need to review this chapter to ensure any SMF considerations are addressed. The target sysplex may differ from the incoming system in the way it switches, manages, and post-processes SMF data.

Moving the incoming system into a PlatinumPlex would result in everything being shared, and jobs typically being able to run on any system in the sysplex. This configuration would allow you to obtain the maximum benefits from being in the sysplex and provide you the greatest flexibility with the management of SMF data. You will need to review this chapter to ensure any SMF considerations are addressed because the target sysplex may differ from the incoming system in the way it switches, manages, and post-processes SMF data.

6.2 Considerations when merging systems into a sysplex

When moving the incoming system into a target sysplex, the SMF considerations will depend on the features you plan to exploit within the sysplex. This section contains, in checklist format, a list of things that must be considered when merging a system into a target sysplex.

Table 6-1 SMF considerations when merging systems into a sysplex

Consideration	Note	Type	Done?
Check for appropriate definitions for SMF data sets	1	B, G, P	

Consideration	Note	Type	Done?
Check how SMF data sets are switched	**2**	**G, P**	
Investigate use of shared SMF parameters	**3**	**G, P**	
Determine how SMF data will be merged and post-processed	**4, 5**	**G, P**	
Check use of User SMF record types	**5**	**G, P**	
Check use of SMF exits	**6**	**G, P**	
Determine SMF records to be collected for each system	**7**	**G, P**	

The “Type” specified in Table 6-1 on page 84 relates to the sysplex target environment—**B** represents a BronzePlex, **G** represents a GoldPlex, and **P** represents a PlatinumPlex.

Notes in Table 6-1 on page 84 are described below:

1. Each system within the target sysplex will need to collect system-specific SMF data in preallocated SMF data sets. You must allocate the data sets on DASD and catalog the data sets. We recommend they be cataloged in the Master Catalog.

   You should have a minimum of two data sets per system for SMF to use, and we recommend that you run with at least three to ensure availability. Select DASD that can handle the volume of data that SMF generates at your installation. If the device is too slow, SMF places the data it generates in buffers. The buffers will eventually fill, which would result in lost SMF data.

   You will also need to consider the size of the SMF data sets. For example, if the incoming system is to be used as a target for DB2 subsystem restarts using ARM or a similar automatic restart mechanism, there may be additional SMF data collected on the incoming system during these times, especially if you use DB2 tracing. For more information about allocating SMF data sets, including VSAM control interval and buffer sizes and so on, refer to Chapter 2, “System Requirements and Considerations”, in *MVS System Management Facilities,* SA22-7630.

   If you plan to have a shared Parmlib configuration, you should consider sharing the SMFPRM member. Having a good naming convention for the SMF data sets will allow you to share the SMFPRM member, as shown in Example 6-1. You will need to allocate the appropriate data sets on the incoming system prior to the merge.

Example 6-1 Example data set specification in the SMFPRM Parmlib member

```
... ... ...
DSNAME(SYS1.&SYSNAME..MAN1,
       SYS1.&SYSNAME..MAN2,
       SYS1.&SYSNAME..MAN3)  /* INDIVIDUAL MANX                */
... ... ...
```

2. When the current recording data set cannot accommodate any more records, the SMF writer routine automatically switches recording from the active SMF data set to an empty SMF data set, and then passes control to the IEFU29 SMF dump exit. A console message is also displayed, informing the operator that the SMF data set needs to be dumped. There are two ways of managing the SMF data set switch:

   a. You could use the IEFU29 dump exit to initiate the SMF dump program, or

   b. You may choose to use automation to trap the console message, and have automation then initiate the SMF dump process.

Either way is valid, but we recommend having a consistent approach for all systems in the target sysplex, especially if you have the IEFU29 exit active in a shared SMFPRM member.

The SMF dump program (IFASMFDP) is used to empty the SMF data sets. It transfers the contents of the full SMF data set to another data set, and resets the status of the dumped data set to empty so that SMF can use it again for recording data. We recommend that you run the SMF dump program *on the system that owns the SMF data sets to be cleared*; do not run the SMF dump program from one system in an attempt to clear a SMF data set used by another system within the target sysplex.

3. The SMF parameters are specified in the SMFPRM member of Parmlib. The 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.

   A shared SMFPRM member can be set up similar to that shown in Example 6-2, using system symbols in the DSNAME and SID parameters.

Example 6-2 Example SMFPRM Parmlib member

```
ACTIVE                      /* ACTIVE SMF RECORDING           */
DSNAME(SYS1.&SYSNAME..MAN1, /* SMF DATA SET NAMES             */
       SYS1.&SYSNAME..MAN2, /* SMF DATA SET NAMES             */
       SYS1.&SYSNAME..MAN3) /* SMF DATA SET NAMES             */
NOPROMPT                    /* NO PROMPT                      */
REC(PERM)                   /* TYPE 17 PERM RECORDS ONLY      */
INTVAL(05)
SYNCVAL(00)
MAXDORM(3000)               /* WRITE AN IDLE BUFFER AFTER 30 MIN*/
STATUS(010000)              /* WRITE SMF STATS AFTER 1 HOUR   */
JWT(0010)                   /* 522 AFTER 10 MINUTES           */
SID(&SYSNAME(1:4))          /* SYSTEM ID IS &SYSNAME          */
LISTDSN                     /* LIST DATA SET STATUS AT IPL    */
LASTDS(MSG)                 /* DEFAULT TO MESSAGE             */
NOBUFFS(MSG)                /* DEFAULT TO MESSAGE             */
SYS(TYPE(30,70:79,89,100,101,110),
EXITS(IEFU83,IEFU84,IEFU85,IEFACTRT,
      IEFUJV,IEFUSI,IEFUJP,IEFUSO,IEFUTL,IEFUAV),
      INTERVAL(SMF,SYNC),NODETAIL)

SUBSYS(STC,EXITS(IEFU29,IEFU83,IEFU84,IEFU85,IEFUJP,IEFUSO,
       IEFACTRT),
       INTERVAL(SMF,SYNC),
       TYPE(30,70:79,89,100,101,110))
```

You will need to review the SMFPRM member prior to sharing the member across systems in the target sysplex to ensure the parameters are appropriate for the incoming system.

For example, you will need to check if the record types or subtypes specified in the SMFPRM member currently being used in the target sysplex are appropriate for the incoming system to ensure that all the required record types are collected. You will also need to check if the exits specified in the SMFPRM member are appropriate or available on the incoming system. If you have differing requirements for SMF parameters between systems, you will need to define a system-specific SMFPRM member in Parmlib.

4. If you already merge SMF data from other systems in the target sysplex, the SMF data from the incoming system should be handled the same way. Merging the SMF data would help with the centralization of report processing, like Workload Licensing Charge (WLC) reporting. Any billing and other post-processing jobs (including performance reporting) on the incoming system may need to be updated to point to the new merged SMF archive data sets.

   You will need to review the SMF data retention period of the target sysplex prior to merging the data from the incoming system to ensure the retention period is appropriate. For example, you don't want to merge data from the incoming system that needs to be kept for seven years, into the target sysplex that has a SMF data retention period of only six months!

   You may also want to look at spinning off the SMF records required for catalog recovery to a set of data sets with a different HLQ and different user catalog than the SMF archive data sets. We recommend this in a shared or non-shared user catalog environment. For more information on catalog backup and recovery, see the IBM Redbook *ICF Catalog Backup and Recovery: A Practical Guide,* SG24-5644.

   You can use the IBM DFSORT™ program product, or its equivalent, to sort and merge SMF data sets from each system into a single dump data set. The MVS System Logger provides a SAMPLIB program, IXGRPT1, to show how you might write a program to analyze the dump data set input and summarize system logger activity across the sysplex. Refer to 6.3.1, “Tools” on page 90 for more information.

5. You will need to ensure that the same user SMF record type is not being used by two different products/applications within the merged sysplex. Record types 128 through 255 are available for user-written records, which could be used by IBM and ISV program products or customer-written applications. For example, if a customer application on the incoming system is writing user SMF record 255 and an ISV product is writing the same record number on the target sysplex there will be issues when merging the SMF data, and subsequently when using common post-processing or reporting programs.

   This is primarily an issue if you merge the SMF data, but we recommend that any clash of record type numbers be addressed when merging the incoming system into the target sysplex to ensure there are no problems in the future. Most ISV products allow you to select the user SMF record number to use.

6. SMF provides exits that allow installations to add installation-written routines to perform additional processing. The SMF exits receive control at different times as a job moves through the system. They receive control when specific events occur, such as when a job CPU-time limit expires. The exit routines could collect additional information, cancel jobs, or enforce installation standards.

   The EXITS parameter within the SMFPRM member of Parmlib specifies which SMF exits are to be invoked, as shown in Example 6-2 on page 86. If an exit is not specified, it is not invoked. If this parameter is not specified, all SMF system exits are invoked. NOEXITS specifies that SMF exits are not invoked. You can specify EXITS on the SYS and SUBSYS statements of SMFPRM. Your choice of SYS or SUBSYS depends on the scope of work you want to influence (system-wide or subsystem-wide), as follows:

   – On the SYS parameter, specifies the exits that are to affect work throughout the system, regardless of the subsystem that processes the work.

– On the SUBSYS parameter, specifies the exits that are to affect work processed by a particular SMF-defined subsystem (JES2, JES3, STC, ASCH, or TSO). The SUBSYS specification overrides the SYS specification.

Ensure the appropriate exits are available with the same, or similar, functionality, by reviewing the function of each of the SMF exits active on the incoming system. This is specifically required if the incoming system will be sharing the sysres with other systems in the target sysplex.

Ensure the EXIT parameters are appropriate for the incoming system by reviewing the SMFPRM member. This needs to be done prior to sharing the member across systems in the target sysplex. If you have differing requirements for SMF parameters between systems, you will need to define a system-specific SMFPRM member in Parmlib.

You can associate multiple exit routines with SMF exits, through the PROG Parmlib member, at IPL, or while the system is running. To define SMF exits to the dynamic exits facility, you must specify the exits in both PROG and SMFPRM. The system does not call SMF exits that are defined to PROG only. If you do not plan to take advantage of the dynamic exits facility, you need only define SMF exits in SMFPRM.

The SMF exits available are:

– **IEFACTRT**: The termination exit. This exit receives control on the normal or abnormal termination of each job-step and job.
– **IEFUAV**: The user account validation exit. This exit is used to validate the accounting information of Transaction Program (TP) users.
– **IEFUJI**: The job initiation exit. This exit receives control before a job on the input queue is selected for initiation.
– **IEFUJP**: The job purge exit. This exit receives control when a job is ready to be purged from the system, after the job has terminated and all SYSOUT output that pertains to the job has been written.
– **IEFUJV**: The job validation exit. This exit receives control before each job control statement (or cataloged procedure) in the input stream is interpreted.
– **IEFUSI**: The step initiation exit. This exit receives control before each job step is started (before allocation).
– **IEFUSO**: The SYSOUT limit exit. This exit receives control when the number of records written to an output data set exceeds the output limit for that data set.
– **IEFUTL**: The time limit exit. This exit receives control when the Job, Step or JWT time limits expire.
– **IEFU29**: The SMF dump exit. This exit receives control when an SMF data set becomes full.
– **IEFU83**: The SMF record exit. This exit receives control before each record is written to the SMF data set.
– **IEFU84**: The SMF record exit. This receives control when the SMF writer routine is branch-entered and is not entered in cross-memory mode.
– **IEFU85**: The SMF record exit. This exit receives control when the SMF writer routine is branch-entered and is entered in cross-memory mode.

You can use the **`D SMF,O`** command to display the current SMF options, as shown in Example 6-3 on page 89, which displays the exits that are active on the system, and whether they are defined at the SYS or SUBSYS level.

Example 6-3 Display SMF options

```
D SMF,O
IEE967I 00.56.16 SMF PARAMETERS 768
        MEMBER = SMFPRM00
        MULCFUNC -- DEFAULT
        MEMLIMIT(00000M) -- DEFAULT
        DDCONS(YES) -- DEFAULT
        DUMPABND(RETRY) -- DEFAULT
        SUBSYS(STC,NODETAIL) -- SYS
        SUBSYS(STC,TYPE(30,70:79,89,100,101,110)) -- PARMLIB
        SUBSYS(STC,INTERVAL(SMF,SYNC)) -- PARMLIB
        SUBSYS(STC,EXITS(IEFACTRT)) -- PARMLIB
        SUBSYS(STC,EXITS(IEFUSO)) -- PARMLIB
        SUBSYS(STC,EXITS(IEFUJP)) -- PARMLIB
        SUBSYS(STC,EXITS(IEFU85)) -- PARMLIB
        SUBSYS(STC,EXITS(IEFU84)) -- PARMLIB
        SUBSYS(STC,EXITS(IEFU83)) -- PARMLIB
        SUBSYS(STC,EXITS(IEFU29)) -- PARMLIB
        SYS(NODETAIL) -- PARMLIB
        SYS(INTERVAL(SMF,SYNC)) -- PARMLIB
        SYS(EXITS(IEFUAV)) -- PARMLIB
        ... ... ...
```

7. To eliminate the collection of duplicate data and help reduce the amount of SMF data collected within a target sysplex, we recommend checking if any SMF records being collected are sysplex-wide. These sysplex-wide SMF records will only need to be collected from one system in the target sysplex. There may be applications or ISV products that write sysplex-wide SMF records, but among IBM products, we were only able to identify the RMF cache data gatherer that did this.

   In a PlatinumPlex configuration, where several systems have access to one and the same storage subsystem, it is sufficient that the cache data gatherer is started just on one system. Running the gatherer on more than one system creates several copies of identical SMF records type 74-5 (Monitor I) or VSAM records (Monitor III). Since RMF has no sysplex control over the gatherer options, it cannot automatically deselect cache gathering on all but one system.

   However, you can achieve this by taking advantage of shared Parmlibs and by the use of system symbols. Define a symbol &CACHEOPT, or similar symbol name, in Parmlib member IEASYMxx (assuming that all the systems in the target sysplex are running in an LPAR) as seen in Example 6-4.

Example 6-4 IEASYM example for RMF CACHE option setting

```
SYSDEF SYMDEF(&CACHEOPT='NOCACHE') /* Global value         */
SYSDEF LPARNAME(PRD1)
       SYMDEF(&CACHEOPT='CACHE')   /* Local value for PROD1 */
```

   Then update or create a shared RMF Parmlib member, ERBRMFxx, with the appropriate symbolic as seen in Example 6-5.

Example 6-5 ERBRMF example for CACHE option setting

```
...                              /* any global RMF parms  */
&CACHEOPT.                       /* CACHE or NOCACHE      */
...                              /* any global RMF parms  */
```

   Then start RMF on all systems in the sysplex using the member option, where xx is the ERBRMF member suffix, as seen in Example 6-6 on page 90.

Example 6-6 Start RMF with member option

```
RO *ALL,S RMF.RMF,,,(MEMBER(xx))
```

With this definition, the symbol &CACHEOPT is defined as “NOCACHE”, while on system PRD1, the symbol is resolved as “CACHE”.

6.3 Tools and documentation

In this section we provide information about tools and documentation that may help you complete this task.

6.3.1 Tools

The following tools may be helpful in managing your SMF data:

SMFSORT

The SMFSORT is sample batch JCL residing in SYS1.SAMPLIB that invokes the SORT program to sort SMF records.

IXGRPT1

System logger provides a program, IXGRPT1 in SYS1.SAMPLIB, to show how you might write a program to analyze the dump data set input and summarize system logger activity across the sysplex. System logger produces SMF record type 88 to record the system logger activity of a single system in a sysplex; these records are written to the active SMF data set on that system. Using the IBM DFSORT program product, or its equivalent, you can sort and merge SMF data sets from each system into a single dump data set. Refer to Chapter 9, “System Logger Accounting” in *MVS System Management Facilities,* SA22-7630 for more information.

ERBSCAN

The ERBSCAN exec resides in hlq.SERBCLS and invokes module ERBMFSCN to scan an SMF data set. The output listing of ERBMFSCN is then displayed in an ISPF EDIT screen. You can then display a single RMF record by entering the command “ERBSHOW recno” in the EDIT command line. In this case, the ERBSHOW exec is invoked as an EDIT macro, which will then re-invoke this exec with the specified record number. Then the corresponding record is formatted (broken into its data sections) and displayed in another EDIT window.

ERBSMFSC

The ERBSMFSC batch JCL resides in SYS1.SAMPLIB and invokes the ERBMFSCN program to scan an SMF data set. The program will list a one-line summary for each SMF record in the input data set.

EDGJSMFP

The EDGJSMFP batch JCL resides in SYS1.SAMPLIB and invokes the ICETOOL program to give a count of each SMF record type found in the SMF data set.

Tivoli Decision Support for OS/390

As a functional replacement for Service Level Reporter (SLR), Tivoli Decision Support for OS/390 allows you to collect and report on systems management data, such as SMF. The MVS component consists of features to collect log data and create reports showing MVS performance characteristics.

Catalog recovery utilities

Catalog recovery utilities, such as IBM's Integrated Catalog Forward Recovery Utility (ICFRU) or Catalog RecoveryPlus from Mainstar Software Corporation (http://www.mainstar.com) use SMF records generated by catalog management to forward-recover catalogs.

6.3.2 Documentation

The following publications provide information that may be helpful in managing your SMF data:

- *ICF Catalog Backup and Recovery: A Practical Guide,* SG24-5644
- *MVS System Management Facilities,* SA22-7630
- *z/OS Resource Measurement Facility (RMF) User's Guide,* SC33-7990

7

JES2 considerations

This chapter discusses the following aspects of moving a JES2 system into a sysplex where all the existing systems are members of the same JES2 Multi-Access Spool (MAS):

- Is it necessary to merge JES2 environments, or is it possible to have more than one JES2 MAS in a single sysplex?
- A checklist of things to consider should you decide to merge the JES2 configurations.
- A methodology for doing the merge.
- A list of tools and documentation that can assist with this exercise.

Note: A JESplex is equivalent to a JES2 multi-access spool (MAS): the set of systems that share a common spool and checkpoint data sets. MAS and JESplex have the same meaning.

7.1 JES2 configurations

In this section, we present the possible JES2 configurations and the possible merging scenarios.

Sharing the spool and JES2 work queues allows you to take advantage of the pooled resources available with multiple MVS systems:

1. Workload distribution.

   A job may be submitted on any member, and be eligible for conversion, execution, output, hardcopy, and purge processing on any member.

2. Workload balancing.

   This is accomplished by using WLM-managed initiators (especially in z/OS 1.4 and later), by setting appropriate class assignments on JES2-managed initiators, and occasionally by using member affinity.

3. Centralized management of output devices.

   You can attach all output devices to a single system, and have that system manage the output for the entire JESplex. Because the devices process the output from all members, they can be run at higher utilizations. And because the output devices select work from all members based on priority and specific work-selection criteria, the highest priority work in the JESplex will be processed accordingly.

4. Simplified operations and maintenance.

   This is achieved by making multiple systems appear as a single system.

In a JES2 MAS configuration, also called a JESplex, all members must be in the same sysplex. Starting with MVS/ESA™ 5.1, JES2 uses XCF to communicate member status and other data between the members of the MAS, so all the systems in the MAS *must* be in the same sysplex. When each JES2 subsystem is initializing, it joins an XCF group. Every member of the MAS must join the same group. The sysplex can be a base sysplex or a Parallel Sysplex.

In a base sysplex, the primary and secondary checkpoint data sets always reside on DASD. An example of base sysplex is shown in Figure 7-1 on page 95, where systems A, B, and C are sharing the checkpoint and spool volumes. System D, which is in the same sysplex, is not in the MAS and has its own checkpoint and spool volumes.

With z/OS, up to four consecutive releases can coexist in a JESplex environment.

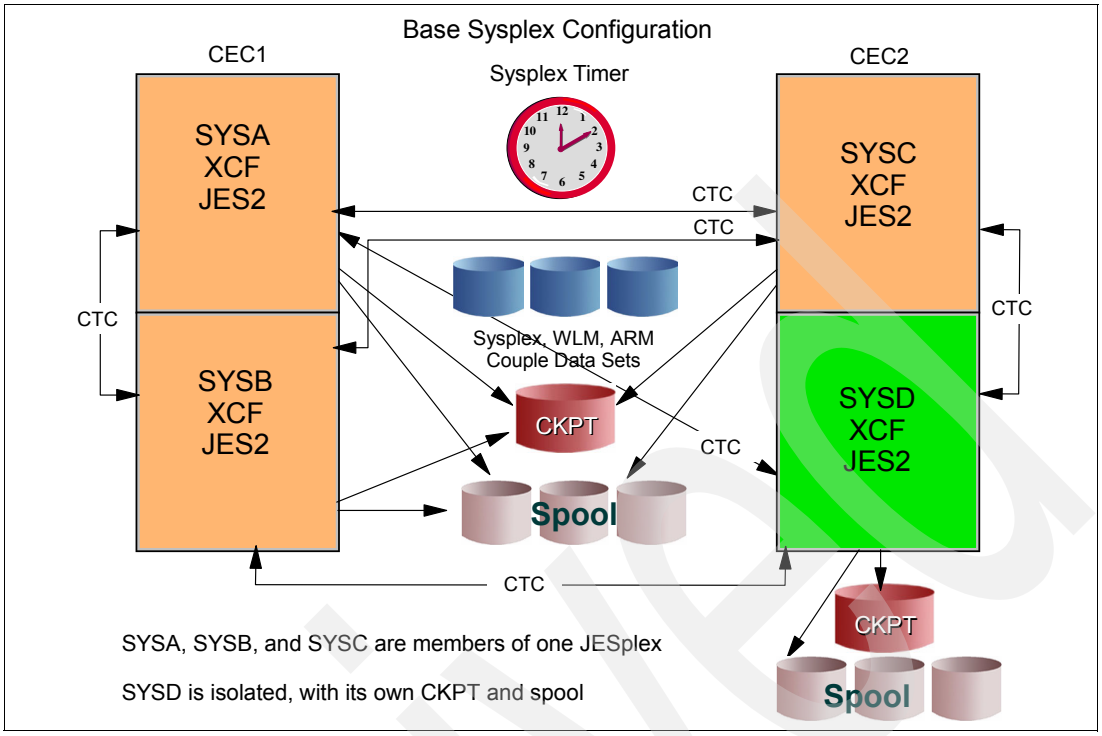


Figure 7-1 JES2 MAS in a base sysplex configuration

Figure 7-2 on page 96 shows another possible target configuration, this time it is a Parallel Sysplex rather than a base sysplex. Once again only three of the systems (SYSA, SYSB, and SYSC) are sharing the spool and JES2 checkpoint data sets. In fact, in this case, the JES2 checkpoints for those three systems are held in the CF. Once again, SYSD is a single JES2 member, not sharing its checkpoint or spool with any other systems.

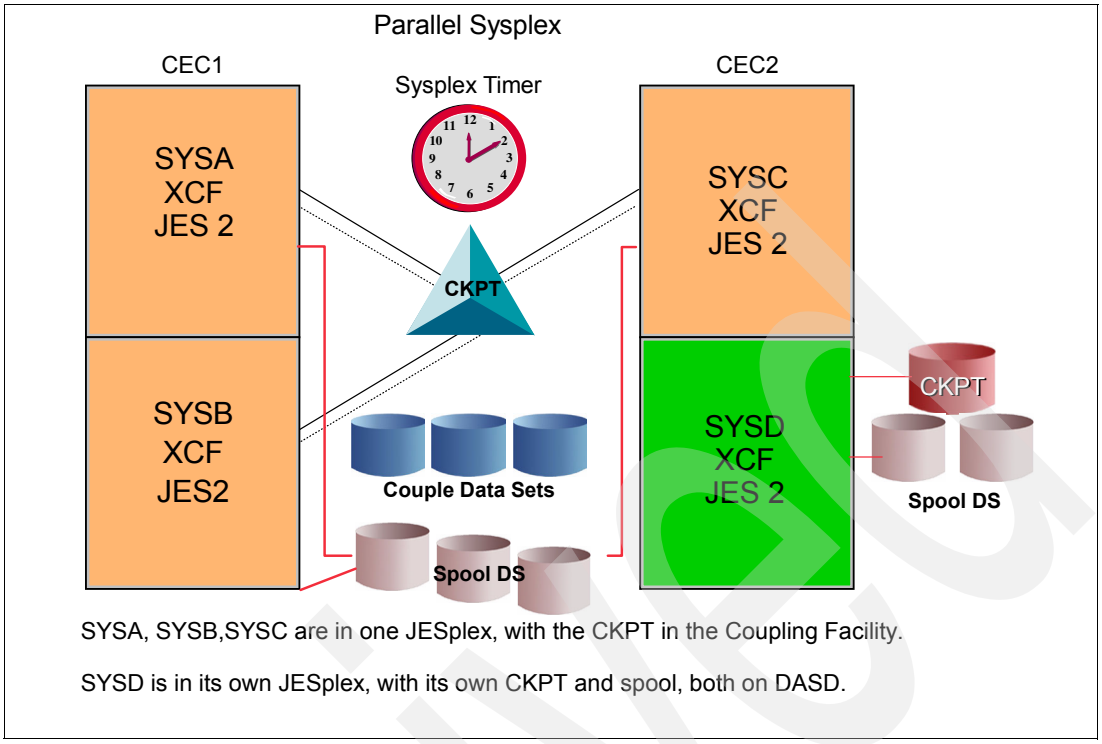


Figure 7-2 JES2 MAS in a Parallel Sysplex configuration

A single member MAS, like SYSD in Figure 7-1 on page 95 and Figure 7-2, can become a multi-member MAS at any time without the need to restart the running system. This has been true since MVS/SP™ 1.3.6. In earlier releases, there were parameters that indicated whether or not JES2 was operating as a MAS or a single system. A cold start was needed to go from non-MAS to MAS operations in those earlier releases.

In this chapter we consider the following possible target configurations, as previously described:

1. BronzePlex: This is a Parallel Sysplex where each JES2 has its own checkpoint and spool data sets, and there is a one-to-one relationship between z/OS systems and JESplexes. This is the easiest to implement, because few (if any) changes are required to the incoming system. However, on an ongoing basis, this is also the most complex configuration to maintain and operate, and it provides no workload balancing or centralized management capability.
2. GoldPlex: This is a Parallel Sysplex with more than one JESplex. The amount of work involved in merging the incoming system depends on whether it will continue to be a single JES member or if it will join one of the existing JESplexes in the sysplex.
3. PlatinumPlex: This is a Parallel Sysplex with only one JESplex, meaning that all systems are sharing the same JES2 checkpoint and spool data sets. This is the ideal scenario. It might involve more work up front to achieve this, but will require less ongoing maintenance and be easier to operate after the merge. It also provides the ability to do workload balancing across all the systems in the sysplex.

The incoming system can come from a configuration where the JES2 is:

1. A single member MAS, with its own checkpoint and spool data sets, or
2. A multi-member MAS, where the whole MAS is being merged with the target sysplex.

7.2 Methodology for merging JESplexes

In this section we discuss the approaches that can be taken to merging JESplexes, and give some advice about the approach that provides the smallest risk or is the easiest to implement.

The steps to follow for merging a system into a Parallel Sysplex depend on what your target configuration is. Before making a decision on what the configuration will be, consider the following points:

- The level of data sharing in the target Parallel Sysplex: for example, will all the DASD be shared between all the systems; will the Master Catalog be shared by all systems; will there be a single RACF database; and so on. The more sharing that you are doing, the more you should try to achieve a single JESplex for the whole sysplex. I

  f, on the other hand, there will be a lot of limitations about where jobs can run, implementing those limitations in a large JESplex can become cumbersome and complex, to the point that it offsets the benefits of having a single JES environment.

- One JESplex can have up to 32 systems; however, as the number of members in the JES2 MAS increases, contention for the JES2 checkpoint also increases. There is a penalty in going from a 1-way to a 2-way MAS. After that, going from a 2-way up to an 8-way there is generally not much of a penalty. The degree of impact depends on the JES2 workload the MAS members are running. Unless the workload is heavy batch and single-threaded submission of jobs, the overhead should be unnoticeable.

  When the number of systems exceeds eight members with JES2-intensive workloads, care must be taken to tune the JES2 parameters, especially HOLD and DORMANCY, and to have a high performance DASD configuration. Also, at this point, the JES2 checkpoint should definitely be in the CF.

  The two biggest problems with large JESplexes are duplicate job name processing and internal reader throughput. The duplicate job name processing problem is only solved by minimizing duplicate jobs, or turning off the restriction on duplicate job names running at the same time (you do this using DUPL_JOB=NODELAY on the JOBDEF initialization statement).

  The internal reader throughput problem is addressed by ensuring that you specify an appropriate number of internal readers, and by ensuring that heavy users of internal readers (such as job scheduling products) are able to utilize multiple readers concurrently. You set the number of internal readers in the RDINUM parameter of the INTRDR statement—prior to z/OS 1.4, the default RDINUM was 2; z/OS 1.4 changed the default to a more realistic 100. There are hints for avoiding internal reader delays provided in the section entitled “Performance Considerations for JES2 Internal Reader”, in *JES2 Initialization and Tuning Guide,* SA22-7532.

  You will find a good discussion about the checkpoint access and configuration modes in the sections entitled “The Checkpoint Cycle” through “Contention-Driven Checkpoint data set Access” in *JES2 Initialization and Tuning Guide,* SA22-7532.

- If the processor speed of the incoming system is significantly faster or slower than the existing members of the target sysplex, you might need to protect the slower members of the MAS from becoming locked out from checkpoint data set access. This is less critical when the checkpoint is in a CF, but you must avoid setting maximum value for DORMANCY too high.

- The alternative to a MAS is Network Job Entry (NJE), which isolates the JES2 MAS complexes but still allows transmission of spool data between them. The considerations for such a configuration are:
  - This provides an isolated environment which can provide protection from other MVS systems.
  - It allows you to isolate workloads and users for security reasons.
  - It reduces spool and checkpoint contention, although it can impose excessive I/O and contention by double-spooling jobs and output that have to be transferred from one MAS to the other.
  - It may be easier to implement initially.
  - It does nothing to decrease the operation and maintenance complexity and workload, in fact it may even result in a more complex and difficult to operate environment.
  - It does nothing to help you balance workload across the sysplex, one of the main reasons for implementing a sysplex in the first place.
  - You need to implement some method (JCL changes, automation, user exits) to ensure the jobs and output end up in the intended MAS.

There are two approaches to moving the incoming system into the target sysplex:

- You can keep the incoming system in its own MAS configuration, like SYSD in Figure 7-2 on page 96. At some later point you may wish to make the incoming system part of the target sysplex JESplex.
- You can merge the incoming system MAS with the target sysplex MAS at the time of the merge.

We assume that the entire MAS that the incoming system is a part of (which might consists of just one system, the incoming system, or it might consist of a number of systems) is being merged into the target sysplex at the same time. While it is possible, we feel it is very unlikely that you would be taking the incoming system *out* of an existing multi-member MAS and moving in *into* the target sysplex on its own, and therefore do not address this scenario.

7.3 Moving to a multi-JESplex environment

In this section we present considerations that you should review if you decide to merge the incoming system into the target sysplex, but with the incoming JES2 retaining its own checkpoint and spool data sets.

Table 7-1 Considerations for merging a system with JES2 isolate

Consideration	Note	Type	Done
Review the tasks to be completed if the incoming system comes from a JE2 MAS	**1**	**B,G**	
Update the MASDEF initialization statement	**2**	**B,G**	
Check use of WLM-Managed Initiators and Scheduling Environments	**3**	**B,G**	
Check use of JES2 exits	**4**	**B,G**	
Determine which JES2 spool data will be moved to the new environment	**5**	**B,G**	

The “Type” specified in Table 7-1 on page 98 relates to the sysplex target environment—**B** represents a BronzePlex, **G** represents a GoldPlex, and **P** represents a PlatinumPlex.

1. Review the JOBCLASS initialization statements:
   a. You can not have the same job class served by both WLM and JES2 initiators. The MODE parameter specifies the initiator type. We recommend that all jobs with the same WLM service class be managed by the same type of initiator (WLM or JES). WLM works more effectively when all initiators servicing a given WLM service class are the same type.
   b. If you are using the SCHENV parameter on the JOBCLASS statement, are the scheduling environments defined to WLM in the target sysplex? For more information about WLM, refer to Chapter 4, “WLM considerations” on page 51.
2. Are JES2 exits being used in the incoming system? Check the EXIT and LOAD statements to identify the exits being used. If exits are being used, are they still needed if the system is going to be in the same sysplex as the target sysplex systems? Do they need changes? Are there new exits that you should add, or new functions to existing exits, because the incoming system is going to be in the same sysplex as the target sysplex systems? It would be a good idea to check the exits and functions of the exits in the existing target sysplex systems to see if any of those functions must be added to the incoming system. As a general rule, however, the fewer exits you use, the better.
3. You need to decide what spool data, if any, that you are going to bring from the old MAS to the new environment. You should test the offload and reload procedures in advance to make sure everything runs smoothly for the actual cutover.

The work you will have in merging your incoming system, from a JES2 perspective, depends on your initial JES2 configuration. If the incoming system comes from a single-member MAS environment and preserves the same configuration after merging, your task will be very easy to implement, with probably nothing to do. If the incoming system is part of a multi-member MAS, the task may be a little more complex, because a multi-member MAS is likely to have a larger workload, more exits, and more customization. However, the basic considerations (from a JES2 perspective) are the same as if you are moving just a single system.

7.4 Moving to a single JES2plex environment

This section contains, in checklist format, a list of things that must be considered should you decide to merge two or more JESplexes. We say JESplexes rather than just JES checkpoint and spool, because there are many more things to be considered than simply moving the incoming system into the same checkpoint and spool as the other systems.

Table 7-2 Considerations for merging JESplexes

Consideration	Note	Type	Done
Decide on exactly what is going to be shared in the target environment	**1**	**G,P**	
Assess the impact of changing the NJE identification of the incoming system	**2**	**G,P**	
Check that there is sufficient spool capacity in the target sysplex	**3**	**G,P**	
Check that the checkpoint data set is large enough for the additional workload	**4**	**G,P**	
If the JES2 checkpoint is in a CF structure, check that it is large enough	**5**	**G,P**	

Consideration	Note	Type	Done
Check the MASDEF parameters	**6**	**G,P**	
Review the performance considerations for checkpoint placement	**7**	**G,P**	
Check if JES2 is in full function or compatibility mode	**8**	**G,P**	
Identify, review, eliminate, or update JES2 exits	**9**	**G,P**	
Decide what to do with JES2-managed devices used by the incoming system	**10**	**G,P**	
Check the use and control of Initiators across the JESplex	**11**	**G,P**	
Check if JES2 job and output class standards are compatible	**12**	**G,P**	
Determine the role of WLM scheduling environments	**13**	**G,P**	
Verify the JES2 initialization parameters that must be the same across the JESplex	**14**	**G,P**	
Review any products that communicate with JES2	**15**	**G,P**	
Review the JES2 start-up procedure	**16**	**G,P**	
Notify the users and system operators		**G,P**	

The “Type” specified in Table 7-2 on page 99 relates to the sysplex target environment—**B** represents a BronzePlex, **G** represents a GoldPlex, and **P** represents a PlatinumPlex. Notes indicated in Table 7-2 on page 99 are described below:

1. Will the systems in the target sysplex and the incoming system or JESplex share:
   a. DASD
      i. To be in a MAS configuration, the SPOOL and checkpoint data sets *must* be shared among all members.
      ii. If some volumes are not shared, you must have a way of preventing jobs that use data sets on those volumes from running in systems not having access to them. Using a WLM scheduling environment is one way to achieve this.
      iii. Related products like DFSMShsm and SMS should also be shared, with the same scope as the DASD that are shared.
   b. RACF database. It would be very unusual to have a single JESplex span more than one RACFplex. If you will *not* have a single RACFplex, you should reconsider whether you really want a single JESplex.
   c. Tape management database, like DFSMSrmm.
   d. Master and user catalogs.

   If the majority of resources are going to be accessible from all systems, then you should definitely merge the incoming system into the target sysplex MAS. On the other hand, if the incoming system will not have access to the resources of the existing systems, and vice versa, then you should probably keep the incoming system in its own JESplex. The one exception may be if the sysplex generates large amounts of JES output—in this case, the benefits of having a single pool of output devices serving all systems may be sufficient to warrant merging all systems into a single JESplex.
2. Each JESplex in a sysplex has a unique node identification. When merging one JESplex into another or an isolated system into a JESPLEX, you have to update all references to the old node id.

Scan all /*XMIT, /*XEQ, or /*ROUTE XEQ (and, in z/OS 1.4 and later, //XMIT) JCL control statements for explicit references to node ids, for example /*XEQ Nx. If the node definitions were identical in both MASes, then you do not have to worry about this. On the other hand, if there are duplicate node numbers pointing to different nodes (for example, in MAS1, N17 points to node POK, while in MAS2, N17 points to node KINGSTON) then you have to update any JCL to work in the target JESplex environment. You can use TSO/ISPF option 3.14 (SUPERC) to find references to the node that is being changed. If there are many to change, you can use the sample CLIST, macro and JCL shown in 7.5, “Tools and documentation” on page 110.

If your installation uses DESTid names for route codes, update DESTID(xxxx) initialization statements with the new node id. It is a good practice to use the node name instead of the (numeric) node id, since it makes changes easier.

Other systems in other sites that have NJE connections to the incoming system must be notified of the node id change, and must change their processes and definitions in synch with the merging of the incoming system into the JESplex.

As an example, let us take 2 nodes, POK and KINGSTON. These 2 nodes are in separate JESPLEXes that are to be merged together. The new node will be POK, and KINGSTON will be obsolete.

When the time comes to change the node, bring down node KINGSTON. This is needed because the **$TNODE** command requires that the node be unconnected before the command will take. On other nodes in the network, issue the **$TNODE(KINGSTON),NAME=newname** command. In this case, **newname** is a name that will never be used as a node in your network.

After issuing this command, issue the command **$TDESTID(KINGSTON),NAME=POK**. This will cause all future output routed to KINGSTON to actually be sent to node POK.

To re-route existing output destined to node KINGSTON to node POK, issue the command **$R ALL,R=newname.*,D=POK.*** For pre-execution jobs, the command is **$R ALL,R=newname,D=POK** . These two commands can be repeated to re-route any output or jobs created using the old node number.

3. Is the current number of spool volumes in the target JESplex adequate to handle the additional workload of the incoming system or JESplex? Additional spool volumes may be required to handle the additional spool files and to better spread the I/O requests and avoid spool contention.

   Also, review the SPOOLNUM parameter of the SPOOLDEF statement, in the JES2 initialization parameters. It specifies the maximum number of spool volumes. The default value is 32; the highest possible value is 253. This value is used when a cold start is done, but subsequent to that, the SPOOLNUM value can only be increased by an operator command (**$T SPOOLDEF,SPOOLNUM=xxx**).

   Just in case a cold start is ever required, you should ensure that the value specified on this parameter in the JES2 initialization parameters always reflects the actual value currently in use.

   All spool volumes in the MAS must be shared among all MAS members. Review the I/O configuration to ensure access to all spool volumes by the incoming system or JESplex.

   Consider if you need to use spool partitioning for the incoming system. Spool partitioning can be a useful tool for increasing JES2 performance. Isolating frequently-run spool-intensive jobs on separate volumes may improve JES2 performance by reducing the load on the other spool volumes. Spool partitioning by job mask can be achieved by the SPOOLDEF initialization statements and two installation exits, Exit 11 and Exit12. Since OS/390 2.10, you can get spool partitioning by system through the use of the command:

   $T SPOOL(nnnnnn),SYSAFF=(sys,sys,...)

For more information about spool partitioning, refer to *JES2 Initialization and Tuning Guide,* SA22-7532 and the IBM Redbook *OS/390 Version 2 Release 10 Implementation,* SG24-5976.

Note: APAR OW49317 provides JES2 support for placing the spool data sets on 3390 volumes containing more than 10019 cylinders per volume. Be aware, however, that the largest spool data set supported is still 64 K tracks.

4. The target checkpoint data set must be large enough to handle the additional workload related to the incoming system or JESplex. Use the $D CKPTSPACE command to determine the amount of free space currently available in the target JES2 checkpoint. If you decide to change some JES2 parameters, be aware of those that affect the checkpoint size, as shown in Table 7-3.

Table 7-3 JES2 Parameters affecting checkpoint size

Initialization parameter	Description	Default value
SPOOLDEF SPOOLNUM=	Number of Spool Volumes	32
SPOOLDEF TGSPACE=(MAX=)	Number of Track Groups on spool	16288
CKPTDEF LOGSIZE=	Size of Change Log on checkpoint	1 (MODE=DUPLEX) 1 to 9 (MODE=DUAL)
CKPTSPACE BERTNUM=	Size of Block Extension Reuse Table	2 x JOBNUM + 100
JOBDEF JOBNUM=	Size of Job Queue	1000
JOBDEF RANGE=	Range of JOBIDs	1–9999
OUTDEF JOENUM=	Size of Job Output Queue	2.5 x JOBNUM

If you need more space, you can use the checkpoint reconfiguration dialog to move the checkpoints to larger data sets or structures. See the section entitled “Operator-initiated Entrance into a Checkpoint Reconfiguration Dialog” in *JES2 Initialization and Tuning Guide,* SA22-7532.

In case of I/O errors or scheduled hardware maintenance, the checkpoint data set can be replaced by the data set specified in the NEWCKPT*n* parameter of the CKPTDEF initialization statement. It can be a CF structure or a data set on DASD. If these data sets are not defined in your JES2 parm member, JES2 will not provide default data sets. However, in that case, you can define these data sets during a checkpoint reconfiguration dialog. If they are specified in the JES2 parms, these data sets are only defined to JES2 and not allocated. The size of these data sets or structures must be at least equal to the size of the primary checkpoint. If you change the checkpoint data set size, remember to also change the NEWCKPT*n* structure or data set size.

Important: The checkpoint data set must be large enough, or JES2 will not initialize.

5. The JES2 checkpoint structure size is a function of the size of the checkpoint data set. If the target sysplex has its checkpoint in a CF and you changed the size of the checkpoint data set, you must review the size of the JES2 structure. When you know how many 4 KB records will be in the new checkpoint data set, you can use the CFsizer tool, available on the Parallel Sysplex home page to calculate the new size of the JES2 structure in the CF:

   http://www.ibm.com/servers/eserver/zseries/cfsizer/jes.html

   Remember that JES2 pre-formats its structure, so it always looks 100% full.

   To find out how to calculate the number of 4 KB records in the checkpoint data set, refer to the HASP537 message issued during JES2 initialization.

Review the size of the structure identified on the NEWCKPT*n* statement. It must be large enough to accommodate the primary checkpoint.

Important: The structure specified on the CKPT1 statement must be sufficient in size, or JES2 will not initialize.

6. The MASDEF statement of the JES2 initialization controls the access to the checkpoint data set through the following parameters:

   HOLD: This specifies the minimum length of time a member of a MAS must maintain control of the checkpoint data set after acquiring it. Setting this value too high needlessly locks out other members in the configuration. Setting it too low causes more delays in local JES2 processing and requires more frequent access (and overhead). Members with a high JES2 workload must hold the checkpoint data set longer than ones with less workload.

   DORMANCY: This specifies both the minimum and the maximum time interval that a member must wait, after releasing the checkpoint data set, before attempting to regain control of it. This parameter can be used to stop a member monopolizing the checkpoint data set by acquiring it too frequently.

   Poor tuning of these parameters can reduce JES2 throughput and can lock out smaller MAS members from access to the checkpoint, or lead to poor JES2 performance on systems with high JES2 workload. A good example is some job scheduler products that control a job's status by issuing JES2 commands. Poor tuning of these parameters can make these products suffer long delays in JES2 environments.

   Also, the communications among JESplex members can be affected by initialization parameters values. When a member of a MAS queues a message to another member of that MAS, there is no interrupt mechanism to inform the receiving member that a message exists. The receiving member periodically reads the shared job queue record and examines queue information for new messages. The length of time it takes for the receiving member to recognize the queuing is controlled by:

   – MASDEF: HOLD and DORMANCY parameters.
   – NJEDEF: DELAY parameter specifies the maximum delay time for intranodal message or command transmission.

   To avoid delays, set the values on the MASDEF statement bearing in mind the profile of the system and its position in the target MAS context. You can use the JES2 command **`$D PERFDATA(QSUSE)`** to display the count and average wait time for $QSUSE requests (access to the checkpoint).

7. In a MAS configuration, the JES2 checkpoint data set is used:
   – To maintain information about job input and output queues
   – For inter-JES2 communications
   – For spool serialization
   – To contain the spool space map

   Review the placement of the primary and secondary checkpoint data sets. Common placements of the checkpoint within a MAS would be:

   a. The Primary checkpoint in a CF structure and the Secondary checkpoint on DASD:

      Placing the primary checkpoint in a CF structure provides more equitable access for all members of a MAS than would be the case if the checkpoint were on DASD. This is because JES2 uses the CF lock to serialize access to the checkpoint. The CF lock is better than the hardware RESERVE and RELEASE macros required for DASD,

because it ensures all members get fair access to the checkpoint through a first-in, first-served queuing mechanism. The CF lock affects only the checkpoint, while the hardware RESERVE macro locks the entire volume upon which the checkpoint data set resides. Also, data can be transferred to and from a CF much faster than to and from DASD.

To ensure the security of that data through the DUPLEX copy retained on DASD, we recommend this placement for MAS configurations with four or more members.

b. Both checkpoints on DASD:

When the checkpoint data set resides on DASD, the change log communicates checkpoint updates to the member that is reading the checkpoint. If the change log is small, it will reside only on the first track of the checkpoint data set.You can use the LOGSIZE= parameter on the CKPTDEF initialization statement to control the size of the change log. Placing both checkpoints on DASD volumes reduces throughput of data and can lock out smaller MAS members from access to the checkpoint. When the primary checkpoint is on DASD, special care must be taken with the I/O configuration and the tuning of the MASDEF HOLD and DORMANCY parameters, since the HOLD time includes the time for JES2 to issue, acquire, and release the RESERVE.

Note: If both checkpoints are on DASD, the best JES2 performance is obtained by running in DUAL mode. However, if the checkpoint is going to be placed in the CF, JES2 must operate in DUPLEX mode. In z/OS 1.2 and later releases, you can switch back and forth between modes using the $T CKPTDEF,MODE=xxxxxx command.

8. The $ACTIVATE command is used to activate new functions at the current release of JES2. The $ACTIVATE command is JESplex-wide and can make JES2 operate in one of two modes:

   a. Full function mode (Z2) or

   b. Compatibility mode (R4)

   Z2 mode was introduced in JES2 z/OS V1R2 and provides constraint relief by increasing the number of JQEs, JOEs, BERTs, and TG space that you can define in the system. This activation level supports the ability of JES2 to process increases in the following:

   a. Job numbers to 999,999

   b. JQEs to a maximum of 200,000

   c. JOEs to a maximum of 500,000

   d. BERTs to a maximum of 500,000

   e. TGSPACE to a maximum of 16,581,181 bytes

   All shared data areas (spool and checkpoint) used by JES2 are compatible with previous releases of JES2. However, the data structures in R4 Mode cannot handle the increased limits. Many binary job number fields are only 2 bytes long and chaining fields use 3 byte offsets which cannot address the new limits. Z2 Mode binary job number fields are 4 bytes long and chaining is accomplished using 3 byte indexes rather than 3 byte offsets. These changes are incompatible with any exit routine or other unique code that examines checkpointed control blocks or processes job numbers.

   If the target JESplex is running in Z2 mode, you may need to:

   a. Update routines in the incoming system that examine JES2 control blocks such as the JQE, where fields are mode sensitive.

   b. Update routines that process job numbers, since they can now have six digits.

c. Contact your vendors for any products based on JES2, running only in the incoming system, to ensure that they are ready to run correctly in Z2 mode.

Important: All members in a MAS configuration in Z2 mode must be on JES2 level z/OS 1.2 or higher. If the target sysplex MAS is already in Z2 mode, but the incoming system is at a lower level than z/OS 1.2, you can revert back to R4 mode using the $ACTIVATE command on the target sysplex MAS. If the target sysplex MAS is still in R4 mode and the incoming system is at a lower level than z/OS 1.2, you should postpone moving to Z2 mode until the incoming system can be upgraded to a Z2-supporting level.

9. JES2 exits are used to satisfy installation-specific needs, such as enforcing job card standardization. JES2 exits allow you to modify JES2 processing without directly affecting JES2 code.

   For operational consistency, it is highly recommended that the exits and the load modules be the same in all members of the JESplex, since a job can run in any member of the JESplex. If they are not the same, unpredictable results can occur.

   To display all enabled exits and the routines associated with them, you can use the JES2 command:

   ```
   $d exit(*),status=enabled,routines
   ```

10. JES2 devices like readers, printers, punches and Remote Job Entry (RJE) workstations may be attachable to any or all members in the JESplex.

    Local devices are attached to the MVS system and are used by JES2 for reading jobs and writing output. Local devices include card readers, card punches, internal readers, and printers. You identify local devices JES2 uses with the RDR(nn), PRT(nn), PUN(nnnn), and LINE(nnnn) initialization statements.

    Review the statements defining local devices on the incoming system to be sure they are unique throughout the target JESplex. You can define the local devices in the same manner to each member in the target MAS. This allows all devices attached to the incoming system to be attached to other members (with appropriate manual switching) when the incoming system is not operational. Similarly, the LINE(nnnn), PRT(nnnn), PUN(nn), and RDR(nn) initialization statements should be set so that a physical device has the same JES2 device number no matter which member it is attached to. For example, suppose there is a 3211 local printer attached to the incoming system. Before merging, the target MAS has 11 local printers defined. The 3211 printer on the incoming system could be defined as:

    ```
    PRT(12) UNIT=1102,START=YES      for the incoming system
    PRT(12) UNIT=1102,START=NO       for the others members of the target JESplex
    ```

    JES2 initialization will detect devices that are not online and place them in a drained state. Later, the device can be activated by entering the $P device and VARY device OFFLINE commands on the member to which it is attached, performing hardware switching, then entering the VARY device ONLINE followed by $S device commands on the new member. The $S command will fail if no hardware path exists.

    If you have defined your own UCSs and FCBs, the SYS1.IMAGELIB data set of the incoming system, make sure that those definitions will be available to the systems in the target sysplex. As part of this, you must check that any UCSs or FCBs with duplicate names are actually identical definitions. If there are, you must address this or else you will only be able to process the output from the incoming system on devices attached to that system.

Review the RJE workstations in the incoming system: be sure that each workstation has a unique subscript number in the target JESplex. For example, if the incoming system has defined RMT(13), RMT(13) should *not* be defined on *any other member of the target JESplex*. RJE workstations can sign on to any member in the MAS, but a given remote number can only be logged/signed on to one member of the MAS at a time.

11. In a MAS configuration, jobs enter the common queue from any input source (local or remote) attached to any member in the configuration. Normally (unless special actions are taken), jobs are eligible to execute in any member in the configuration. Started tasks and TSO/E users are an exception: they execute only in the member in which they are entered. As the members logically share common JES2 job input and output queues, the workload can be balanced among members by allowing jobs to execute on whatever member has an idle initiator with the correct setup and required output devices. There are two different modes of initiators:

    a. Those owned by JES2 and controlled by JES2 commands.

    b. Those owned and managed by WLM.

    The MODE parameter on the JOBCLASS statement determines whether jobs in that class will be selected by a JES2 or WLM initiator, *and applies to all systems in the MAS*. Initiators from each JES2 member in the MAS select work from the common job queue independently of other initiators on other members of the JESplex.

    Review the JOBCLASS initialization statements:

    – In the incoming system, are there WLM-Managed initiators? If so, make sure they are defined for the same classes as the WLM-Managed initiators (if any) in the target sysplex. If a given job class is managed by WLM in one environment and by JES2 in the other, then you need to decide how you are going to proceed. The JOBCLASS definitions are MAS-wide, so you cannot have a class managed by WLM on one system and managed by JES2 on others.

    In addition to considerations about WLM- and JES-managed initiators, you also need to consider:

    – Which job classes will the JES-managed initiators on the incoming system run?

      You need to review the classes that those initiators handle prior to the merge, and the jobs that you want this system to run after the merge. It is likely that, unless you take some specific action to prevent it, the initiators on the incoming system will start running jobs from the other systems after the merge. If this is true, you need to make sure that those jobs can execute successfully on the incoming system: for example, does JES2 on the incoming system contain all the required user Proclibs, does the incoming system have access to all the required devices (both DASD and tape), will any required database managers be available, and so on.

    – Where will the jobs submitted from the incoming system run?

      If the incoming system is going to run jobs from other systems in the MAS, you need to make sure that all the jobs that previously only executed on the incoming system will continue to be processed in a timely manner by one of the systems in the MAS, and that all candidate systems have access to all required devices, Proclibs, database managers, and so on.

12. Is the target JESplex using standards for job and output classes? If so, is the incoming system using the same standards? For example, do jobs that require a manual tape drive use job class T on the incoming system, but class M on the existing systems in the target sysplex? If the standards are not the same, you need to decide how to proceed. If the standards are widely accepted *and used* (which is not always the case!), it can be a sizable task to change them. Products like ThruPut Manager from MVS Solutions can help by providing a high level way of defining rules for what jobs should be allowed to run where

within the MAS. More information about ThruPut Manager can be found on the MVS Solutions Web site at:

http://www.mvssol.com/

13. As soon as the incoming system joins the MAS, any job can potentially run in any member of the MAS. In a sysplex environment, there can be resources that are only accessible to a subset of the systems. In a multi-member MAS environment, jobs using these resources must execute on a system that has access to the required resources. There are a number of ways of directing jobs to a particular system or systems:

    a. You can use job classes to associate a job with certain resources and control which classes the initiators on each system can select; see the discussion in item 12 on page 106. Using this mechanism, you can potentially have a number of systems that are candidates to run a given job.

    b. Batch jobs can specify which members in the MAS can process them by explicitly coding the member names on the SYSAFF parameter of the /*JOBPARM statement in their JCL. If you use this mechanism, the list of candidate systems that a job can run on is hardcoded in the job's JCL, which is not really an ideal situation.

    c. A job may be assigned a scheduling environment to ensure it executes on the members in the MAS which have the required resources or specific environmental states. This mechanism provides more flexibility and granularity than either of the previous mechanisms. However, like option b, it requires JCL changes or the implementation of a JES exit to assign the appropriate scheduling environment to the job.

    d. You can use a non-IBM product such as ThruPut Manager. This provides the most flexibility and power; however, unlike the other three options, there is a financial cost associated with this option.

    The scheduling environment can be specified with the SCHENV= keyword parameter on the JOB statement. It can also be assigned via the JES2 JOBCLASS initialization statement. This means that every job that is submitted in a particular job class *from any system in the MAS* will be assigned the same scheduling environment.

    Scheduling environments must be defined to WLM. The scheduling environment is validity-checked by the converter and, if not defined to WLM, the job will fail with a JCL error.

    JES2 detects the availability of scheduling environments on each member and allows an initiator to select jobs only if the specified scheduling environment is available. This is true for both JES2-managed initiators and WLM-Managed initiators.

    If you decide to use this mechanism, or are already doing so, you must review the scheduling environments defined in the WLM policies in the target Parallel Sysplex:

    – You may need to add new resources and scheduling environments to fit the incoming system needs.

    – If the incoming system was already using scheduling environments, look for different scheduling environments using the same name, and so on.

    If you have to change the name of a scheduling environment, remember that any JCL that used the old scheduling environment name will need to be updated. For more information about merging WLM policies, see Chapter 4, "WLM considerations" on page 51.

    This may present a good opportunity for installations using batch-scheduling products to replace the resource definition in those products with the WLM resource definition and scheduling environments.

    You can display all the scheduling environments defined to WLM, and on which members in the MAS each is available, by using the SDSF "SE" option.

14. The parameters that must be the same in the JES2 initialization across all members in the MAS are as follows:

xxxDEF Global JES2 parameters, but some particularities are listed below.

MASDEF DORMANCY and HOLD values can vary according to JES2 workload and response requirements. We recommend letting the OWNMEMB default to the SMFPRMxx SID value.

NJEDEF OWNNODE and NODENUM must be the same on all members and the local node definition parameters as specified on the NODE(nnnn) initialization statement must be identical. If JES2 starts with other MAS-members already active, and the information in NODENUM differs from the MAS node number, the JES2 start is aborted and the following messages are issued:

```
$HASP442 INITIALIZATION STATEMENTS CONFLICTING WITH SAVED VALUES FOLLOW:
$HASP496 NJEDEF NODENUM=38 CANNOT BE CHANGED FROM SAVED VALUE OF 25 AND MUST BE
CORRECTED TO START JES2
$HASP428 CORRECT THE ABOVE PROBLEMS AND RESTART JES2
```

CONDEF This defines the JES2 console communication environment. These are usually set the same on each member, but if SCOPE=SYSPLEX is specified, the CONCHAR parameter must be unique on each of the MAS members.

APPL This defines the VTAM application name for each node with which an installation plans to connect. If the APPL(avvvvvvv) provides an incorrect node name (NODE=), line (LINE=) or compaction table (COMPACT=), installations can often establish a VTAM session, but are unable to activate the NJE session.

BADTRACK This specifies a range of track addresses of defective tracks on a specific spool volume. JES2 will not attempt to use tracks identified as defective. So, as the members are using the same spool, this information must be consistent in all MAS-members.

CONNECT This specifies static connections between nodes. For consistency, they must be the same.

DESTID(xxxx) This specifies an installation-defined name for a JES2 route code. By defining DEStid names for route codes, the users and operators can refer to the DEStid names instead of the explicit names provided by JES2. It must be consistent throughout the MAS-members.

ESTBYTE This specifies, in thousands of bytes, the default estimated output (SYSOUT) for a job at which the `BYTES EXCEEDED` message is issued, and the subsequent action taken. To avoid different behavior throughout the MAS members, use the same value in all members.

ESTIME This specifies the default elapsed estimated execution time for a job, the interval at which the `TIME EXCEEDED` message is issued, and whether the elapsed time job monitor feature is supported. For consistent behavior in the JESplex, its values must be the same for all MAS members.

ESTLNCT This specifies the default estimated print lines output for a job, the interval at which the `LINES EXCEEDED` message is issued, and the subsequent action taken. Since a job can be executed in any MAS-member, for consistent behavior it must be the same in all members.

ESTPAGE The same as ESTLNCT, but for pages.

ESTPUN The same as ESTLNCT, but for cards.

EXIT Different exits used in different JES2s do not provide a consistent treatment across the JESplex, since jobs can be executed in any MAS-member. The exits used, and the functions of the exits, should be identical on all systems in the MAS.

INTRDR For consistent job behavior throughout the MAS, INTRDR should have identical values on all systems. As an example, one of the parameters authorizes certain JES2 commands to be submitted through an internal reader. If that information is different between MAS members, then the same job, issuing the same JES2 commands, can have the commands accepted in one member but not in another.

JOBCLASS For consistent job behavior throughout the MAS, the JOBCLASS definition should be the same in all members. The JOBCLASS statement is read by JES2 when you do a cold start, *and* during certain recovery processing; it is not read during warm start processing. Normally changes to JOBCLASS definitions are implemented using operator commands; however, you should always remember to reflect permanent changes back to the initialization members used by all systems in the MAS.

JOBPRTY This specifies the job-scheduling priority relationship with execution times. For consistent behavior, they should be the same.

LOADMOD For consistency across the JESplex, each exit should use the same load module version.

NETACCT This specifies a network account number and an associated local account number. JES2 uses the parameters on the NETACCT statement to build the network account table used by HASPRDR, the job and SYSOUT transmitters, and the job and SYSOUT receivers. For a consistent table in all members, that value should be the same.

NODE(xxxx) This specifies the characteristics of the node to be defined. For operational consistency, it must be identical in all MAS-members. As an example, JES2 can use the node name specified for OWNNODE to determine the JES2 XCF group member name.

OUTCLASS This specifies the SYSOUT class characteristics for a specific output class. The values must be consistent throughout the sysplex to avoid different handling if a job runs in one member or another.

15. There are many products that interact with JES2, such as JES/328X, INFOPRINT, JESPRINT, SDSF, CA-SPOOL (Computer Associates product), CONTROL-M (BMC product) and others. Prior to the merge, review their installation, customization, and maintenance procedures to ensure they will continue to function when the incoming system becomes part of the MAS.

16. Jobs can be converted in any member. In order to ensure successful conversion, all member start-up procedures should use the same data sets and concatenation order. Also, the Proclib initialization statement should be consistent across all members. Proclibs defined using the JES2 Proclib initialization statement can be displayed, updated or deleted using operator commands without restarting JES2.

7.4.1 SDSF considerations

SDSF customization is performed using ISFPARMS, which defines global options and the format of the panels. By using the options, you can choose whether or not SDSF should use JES2 checkpoint versioning; which groups of users and the SDSF functions will be available to members of the group; and so on. SDSF provides two alternatives for customizing ISFPARMS: a set of macros that are assembled and link-edited, or statements that are read by the SDSF Server address space. When merging the incoming system into the target sysplex, you should review the items in Table 7-4.

Table 7-4 SDSF considerations

Consideration	Note	Sysplex	Done
SYSNAME(xxxx) in WHEN statement, if used	1	G, P	
GROUP or ISFGRP macro	2	G, P	
DESTDEF JES2 initialization	3	G, P	

1. The WHEN statement can be used to conditionally process an entire ISFPARMS statement (OPTIONS, GROUP, and so on) based on the name of the system the user is logged on to. If you use this facility, you may wish to update your ISFPARMS to add the incoming system, and customize accordingly.
2. An ISFGRP macro or GROUP statement defines a group of users and specifies which functions the members of the group may perform. Review this if you need to update whether the incoming system shares a common ISFPARMS with the other members of the target sysplex.

   If you are using SAF along with your group definitions to control membership and authorization, review if you need to update the definitions, as you may want some users to have access to other members of the MAS.
3. The SHOWUSER parameter of the JES2 DESTDEF initialization statement affects how destinations are presented to SDSF. This parameter can affect SDSF security profiles. If DESTDEF SHOWUSER=WITHLOCAL is coded, then destinations of the form *local-node.userid*, which are otherwise displayed as userid, are displayed as *LOCAL.userid*. If you changed the field list definitions for the PR display and you coded a default width for the destination column in the ISFFLD macro or FLD statement (that is a width of “D”), then the length of the column will be 18 rather than 8 to accommodate the longer destination name that will be displayed.

7.5 Tools and documentation

In this section we provide information about tools and documentation that may help you complete the merging task.

For searching partitioned data sets, you can use either TSO/ISPF option 3.14, Search-For, or option 3.15, Search-ForE. If you need to change data in many members of a PDS data set, you can use ISPF ISREDIT macros that are very easy to code. Example 7-1 contains a sample CLIST that allows you to make mass changes when used with the EDIT macro contained in Example 7-2 on page 111.

Example 7-1 Sample CLIST for member processing

```
PROC 0 DSN()
/*-------------------------------------------------*/
/* CHANGE:  CLIST_NAME  TO THE CLIST MEMBER NAME    */
```

```
/*          MYCHANGE    TO THE MACRO MEMBER NAME    */
/* FOREGROUND CALL TO CLIST:                        */
/* CLIST_MEMBER_NAME DSN(DATA_SET_NAME_TO_SEARCH)    */
/*--------------------------------------------------*/
SET RC = 0
IF &DSN =  THEN +
   DO
   WRITE '*** INCORRECT CALL TO CLIST CLIST_NAME ***'
   WRITE '*** CORRECT CALL IS: CLIST_NAME DSN(DSN_TO_PROCESS) ***'
   EXIT CODE(50)
   END
ISPEXEC LMINIT DATAID(TEMPDDN) DATASET('&DSN') ENQ(SHR)
SET LMRC = &LASTCC
IF &LMRC NE 0 THEN +
   DO
   WRITE '*** LMINIT ERROR RC=&LMRC ***'
   EXIT CODE(&LMRC)
   END
ISPEXEC LMOPEN DATAID(&TEMPDDN) OPTION(INPUT)
SET LMRC = &LASTCC
IF &LMRC NE 0 THEN +
   DO
   WRITE '*** LMOPEN ERROR RC=&LMRC ***'
   ISPEXEC LMFREE DATAID(&TEMPDDN)
   EXIT CODE(&LMRC)
   END

SET MEMBER =
SET LMRC = 0
/* LOOP TO ALL MEMBERS OF THE PDS */
DO WHILE &LMRC = 0
ISPEXEC LMMLIST DATAID(&TEMPDDN) OPTION(LIST) MEMBER(MEMBER)
SET LMRC = &LASTCC
IF &LMRC = 0 THEN
   DO
   ISPEXEC EDIT DATAID(&TEMPDDN) MEMBER(&MEMBER) MACRO(MYCHANGE)
   IF &LASTCC > 4 THEN +
      DO
      WRITE 'ERROR PROCESSING MEMBER &MEMBER'
      SET RC = 12
      SET LMRC=8
      END
   END
END
/*-------------------------------------------------------------*/
/* FREE THE MEMBER LIST, CLOSE AND FREE THE DATAID FOR THE PDS. */
/*-------------------------------------------------------------*/
ISPEXEC LMMLIST DATAID(&TEMPDDN) OPTION(FREE)
ISPEXEC LMCLOSE DATAID(&TEMPDDN)
ISPEXEC LMFREE DATAID(&TEMPDDN)
EXIT CODE(&RC)
```

Example 7-2 contains a sample EDIT macro to help you make mass changes.

Example 7-2 Sample MYCHANGE EDIT macro to find and change strings

```
ISREDIT MACRO
/* CHANGE OLD_STRING TO THE STRING YOU ARE SEARCHING FOR */
/*        NEW_STRING TO THE STRING TO THAT WILL REPLACE OLD_STRING */
ISREDIT X ALL
```

```
SET LCC = 0
ISREDIT FIND 'OLD_STRING' ALL WORD
IF &LASTCC = 0 THEN +
   DO
   ISREDIT (MEMNAME) = MEMBER
   ISREDIT CHANGE 'OLD_STRING' 'NEW_STRING' ALL NX
   ISREDIT SAVE
   SET LCC = &LASTCC
   IF &LCC =  0 THEN +
      WRITE *** &MEMNAME CHANGED ***
   ELSE DO
      IF &LCC = 12 THEN +
         WRITE &MEMNAME  NOT SAVED NOT ENOUGH PDS OR DIRECTORY SPACE
      ELSE
         WRITE  &MEMNAME NOT SAVED - RC=&LCC
      ISREDIT CANCEL
      END
   END
ISREDIT END
 EXIT CODE(&LCC)
```

You can execute the sample CLIST through TSO/ISPF Option 6, or you can use the sample JCL contained in Example 7-3 to invoke the CLIST from a batch job. If you wish to modify the sample macro, refer to *z/OS V1R2.O-V1R3.0 ISPF Edit and EDIT Macros,* SC34-4820.

Example 7-3 Sample JCL to invoke CLIST from batch

```
//SEARCH EXEC PGM=IKJEFT1A
//SYSPROC DD DISP=SHR,DSN=PDS_CONTAINING_CLIST_AND_MACRO
//ISPPLIB  DD DISP=SHR,DSN=ISP.SISPPENU
//ISPMLIB  DD DISP=SHR,DSN=ISP.SISPMENU
//ISPSLIB  DD DISP=SHR,DSN=ISP.SISPSENU
//ISPTLIB  DD DISP=SHR,DSN=ISP.SISPTENU
//ISPLOG   DD SYSOUT=*,LRECL=125,RECFM=VBA
//ISPPROF  DD DISP=(,DELETE),DSN=&&ISPPROF,LRECL=80,DSORG=PO,
//        SPACE=(TRK,(5,5,2)),RECFM=FB,BLKSIZE=0
//SYSTSIN  DD *
  ISPSTART CMD(CLIST DSN(DSN_TO_SEARCH)) BDISPMAX(99999)
/*
```

To determine the size of the JES2 checkpoint structure, you should use the CFSizer wizard available on the Parallel Sysplex home page at:

http://www.s390.ibm.com/cfsizer/jes.html

You can find a good description of JES2 processing in a sysplex configuration in the IBM Redbook *OS/390 Parallel Sysplex Configuration Volume 2: Cookbook,* SG24-5638.

For a valuable source of information about initialization and tuning refer to *JES2 Initialization and Tuning Guide,* SA22-7532.

For specific information about processing in a MAS, refer to Washington System Center Technical Bulletin *JES2 Multi-Access Spool in a Sysplex Environment,* GG66-3263.

For a good explanation of the JES2 enhancements relating to I/O to spool and spool partitioning, refer to the IBM Redbook *OS/390 Version 2 Release 10 Implementation,* SG24-5976.

After merging the incoming system into a JESplex, you should monitor the performance of JES2. Realistically, the best measurement tool for your JES2 checkpoint is the lack of symptoms of JES2 delays by your applications! However, here are some tools you can use for checkpoint analysis:

- SDSF: The MAS option displays the members status, hold and dormancy times and actual times. This also a convenient panel for adjusting the times and seeing immediate results. Be aware, however, that these times are only instantaneous, and do not show averages.
- RMF III: See the Subsystem Display, then “JES Delays”. Excessive delays here are often due to checkpoint delays.
- RMF CF Structure Activity Report: JES2 writes many blocks of data at once, so you will often see “No Subchannel Available” in these reports. This is normal and should not alarm you. The service times for Synch and Asynch requests should be within the published guidelines for your environment.
- $D PERFDATA(QSUSE): This displays the count and average wait time for $QSUSE requests (access to the checkpoint). You will find the description and instructions about how to use the $PERFDATA JES2 command in:

  http://www.ibm.com/support/techdocs/atsmastr.nsf/PubAllNum/W9744B

- $TRACE(17) data: Turn on $TRACE(17) records for ten to fifteen minutes during your most active time of the day (from a JES2 perspective). Then analyze the data with the JES2T17A sample program provided with JES2. Here are sample JES2 operator commands to trace to class x:

  ```
  $S TRACE(17)
  $T TRACEDEF,TABLES=20
  $T TRACEDEF,ACTIVE=Y,LOG=(START=Y,CLASS=x,SIZE=92000)
  ```

  When you have collected sufficient data, spin off the $TRCLOG, and turn off tracing, then use the IBM external writer (XWTR) or SDSF to write the trace records to disk:

  ```
  $T TRACEDEF,ACTIVE=NO,SPIN
  $P TRACE(17)
  S XWTR.X,UNIT=SYSDA,DSNAME=JES2.TROUT,SPACE=(CYL,(1,3))
  F X,CLASS=x
  ```

  For a description about JES2 trace records available, refer to *z/OS V1R3.0 JES2 Diagnosis*, GA22-7531.

8

Shared HFS considerations

This chapter discusses the following aspects of moving a system that is using its own HFS environment into a sysplex where all the existing systems share some or all of the HFSs:

- Is it necessary to merge HFS environments, or is it possible to have more than one HFS environment in a single sysplex?
- A checklist of things to consider should you decide to merge the HFSs.
- A methodology for doing the merge.
- A list of tools and documentation that can assist with this exercise.

8.1 One or more HFSplexes

An HFSplex is a collection of z/OS systems that share the OMVS CDS and sysplex root HFS, and have SYSPLEX(YES) specified in their BPXPRMxx member. The CDS contains the sysplex-wide mount table and information about all participating systems, and all mounted file systems in the HFSplex. It is *not* possible to have more than one HFSplex in a single sysplex.

Prior to OS/390 2.9, it was not possible for more than one system to have read/write access to an HFS at a time. It *was* possible for multiple systems to have the HFS mounted read only, but in that case, *no* one could have it mounted R/W, and all the sharing systems had to be within the same GRS complex—this is because ENQs were used to ensure no one mounted the HFS read/write if it was already mounted read-only on another system. As a result of this restriction, it is unlikely that the incoming system had any access to the target sysplex HFSs prior to the merge and vice versa. Therefore, this chapter is written on the assumption that each of the environments (incoming system and target sysplex) had their own dedicated HFSs prior to the merge.

Note that it is still possible to share HFSs in read/only mode in OS/390 2.9 and later systems, in addition to the new read/write file sharing provided with OS/390 2.9 and later systems.

There is more information about both read-only and read/write HFS sharing in the IBM Redbook *Hierarchical File System Usage Guide,* SG24-5482.

In line with the sysplex target environment options outlined in 1.2, “Starting and ending points” on page 2, here are the definitions of the possible scenarios:

- Moving the incoming system into a BronzePlex would typically be done to obtain the benefits of PSLC or WLC, and therefore would have minimal sharing. There is no requirement for sysplex-wide HFS sharing within the target sysplex in a BronzePlex configuration, and you are unlikely to want to share HFSs in this configuration.

  In addition, you should note that you can only have one HFSplex per sysplex. While it is possible to use sysplex HFS sharing to share a set of HFSs between the systems in one subplex, and not use sysplex HFS sharing in the systems in the other subplex, you cannot use sysplex HFS sharing to share one set of HFSs between all the systems in one subplex (and only those systems) and also use sysplex HFS sharing to share a different set of HFSs between all the systems in another subplex in the same sysplex.
- Moving the incoming system into a GoldPlex would typically be done to share the system software environment (sysres), plus other selected system infrastructure elements. There is no requirement for sysplex-wide HFS sharing within the target sysplex in a GoldPlex configuration, however, it is possible.

  A GoldPlex has the same considerations as a BronzePlex in terms of not being able to have more than one HFSplex per sysplex.

  A GoldPlex might not include the sharing of the Master Catalog within the target sysplex and therefore user catalogs may or may not be shared. So, you will need to consider how you want to manage the HFS data set catalog entries. If the HFS data sets are cataloged in the Master Catalog and you have more than one Master Catalog (one per system, for example), you will need to define a process to keep the Master Catalogs in sync across the sysplex. Another alternative is to have the HFS data sets in a user catalog that is shared by all systems in the sysplex.

  A GoldPlex can be set up to share the version HFS data sets on the shared sysres without needing to set up sysplex-wide HFS sharing. However, if you wish to use the sysplex root HFS from the incoming system, and the target sysplex is using a sysplex-wide root HFS, you must merge the incoming system into the HFSplex.

Access to the internal file and directory structure of a shared HFS is controlled by the UNIX security, which is at the “owner”, “group”, or “other” level. So, systems sharing HFSs will see the same “owner”, “group”, “other” flags across the HFSplex. The files or directories in UNIX have “owner” and “group” entries associated with each. The “owner” and “group” are numbers (UID and GID) associated with the RACF userids or groups (defined in the OMVS segment of each). If you do not share the RACF data bases within the target sysplex, but you do share the system HFS data sets, then the userids and groups need to have the same UIDs and GIDs respectively across the RACFplex.

For example, if userid “Fred” has a UID of 1234 on system TEST and system DEVL shares the HFSplex with TEST, but not RACF, then userid “Fred” needs to have a UID of 1234 on system DEVL too, to ensure the “owner”, “group”, “other” flags are honoured correctly. If userid “Jane” has UID 1234 on DEVL instead, then Jane would have the same access as “Fred” to files and directories in the HFSplex.

Given the difficulty that would be involved in maintaining UIDs in this manual way, we do not recommend placing the incoming system in the same HFSplex as the target sysplex unless the incoming system is also in the same RACFplex as the target sysplex.

- Moving the incoming system into a PlatinumPlex means sharing everything, including the system infrastructure, user catalogs, and all user volumes. This configuration allows you to gain the benefits of sharing not just the sysplex-wide root HFS, but also full read/write sharing of all user HFSs from any system within the sysplex. There are a number of advantages when moving into a PlatinumPlex configuration and using sysplex HFS sharing.
- Every HFS data set that is shared read/write in the sysplex has a central owning system that manages it on behalf of other systems in the sysplex. The owning system gets read or write requests from other systems and performs these operations on the HFS data sets. A messaging protocol using XCF services is used to transfer data around the sysplex from the central owner. When HFS data sets are shared across the sysplex, you will have a single large pool of all the HFS data sets of the entire sysplex. All these HFS data sets will be fully accessible from every system in the sysplex.

Important: It is vital to understand that once a system joins a HFSplex, *all* the HFSs on that system, and *all* the HFSs on all the other systems in the HFSplex are accessible to *every* system in the HFSplex, *even if the volume containing a given HFS is not online to some of the systems*. As long as a volume is online to just one system, every system in the HFSplex can access the HFSs on that volume.

8.2 Considerations for merging HFSplexes

This section contains, in checklist format, a list of things that must be considered should you decide to merge two or more HFSplexes.

Table 8-1 Considerations for merging HFSplexes

Consideration	Note	Applies to	Done
Create the sysplex root data set	**1**	**G, P**	
Create system-specific HFS data sets	**2**	**G, P**	
Create OMVS CDSs	**3**	**G, P**	
Update COUPLExx to define the OMVS CDS to XCF	**3**	**G, P**	
Handling multiple version HFS data sets	**4**	**G, P**	

Consideration	Note	Applies to	Done
Check the MOUNT attribute for the version HFS	**4**	**G, P**	
Customize BPXPRMxx for HFS sharing	**5**	**G, P**	
Check UNIX System Services security (HFS)	**6**	**G, P**	
Verify SMS ACS routines for HFS user data sets	**7**	**G, P**	
Considerations for using zFS in a sysplex in shared HFS mode	**8**	**G, P**	
Review other considerations for z/OS UNIX	**9**	**G, P**	

The “Type” specified in Table 8-1 on page 117 relates to the sysplex target environment—**B** represents a BronzePlex, **G** represents a GoldPlex, and **P** represents a PlatinumPlex. Notes indicated in Table 8-1 on page 117 are described below:

1. The sysplex root is an HFS data set that is used as the sysplex-wide root. This data set is required for the sysplex HFS sharing capability provided in OS/390 2.9, and is only used if that capability is being used. It provides an umbrella structure that encompasses all the HFS files in the sysplex. It contains directories and symlinks to redirect HFS file references to the correct files across the sysplex. It does not contain any files or any code. The sysplex root has symlinks to point to /bin, /usr/, /lib and /opt files that are specific to a release/service level of z/OS and /dev, /tmp, /var, and /etc files that are specific to each system in the sysplex. For more information refer to Chapter 18, “Shared HFS in a sysplex”, in *z/OS UNIX System Services,* GA22-7800.

   When a system mounts the sysplex root HFS, a mount point for that system is dynamically added in that HFS. For this reason, the sysplex HFS data set must be mounted read-write and designated AUTOMOVE. For more information about AUTOMOVE, see *z/OS UNIX System Services,* GA22-7800.

   You will need to ensure sysplex HFS sharing is enabled on all systems within the HFSplex by specifying the SYSPLEX(YES) parameter in the BPXPRMxx member of Parmlib, as in Example 8-1. Note that this parameter is only available on systems at OS/390 2.9 and above. Also see Example 8-4 on page 122 for more information on the BPXPRMxx filesystem definitions.

Example 8-1 Enable sysplex HFS sharing in BPXPRMxx

```
... ... ...
SYSPLEX(YES)                    /* sysplex enabled              */
... ... ...
```

   Only one sysplex root is allowed for all systems participating in an HFSplex. The sysplex root is created by running the BPXISYSR sample job in SYS1.SAMPLIB. After the job runs, the sysplex root file system structure would look like the one in Example 8-2 on page 119.

   You will notice the $VERSION and $SYSNAME symbolic links. A directory with the value specified on the VERSION parameter of BPXPRMxx will be dynamically created at system initialization under the sysplex root and will be used as a mount point for the version HFS.

   If the content of the symbolic link begins with $SYSNAME and SYSPLEX(YES) is specified in BPXPRMxx, then $SYSNAME is replaced with the system name when the symbolic link is resolved, which is used for system-specific HFS data sets. The presence of symbolic links is transparent to the user.

Example 8-2 Sysplex root file system structure

```
Sysplex Root
...
bin =====> $VERSION/bin
usr =====> $VERSION/usr
lib =====> $VERSION/lib
opt =====> $VERSION/opt
samples => $VERSION/samples
$VERSION ================> $VERSION/
$SYSNAME ================> $SYSNAME/
dev =====> $SYSNAME/dev
tmp =====> $SYSNAME/tmp
var =====> $SYSNAME/var
etc =====> $SYSNAME/etc
u
```

No files or code reside in the sysplex root data set. It consists of directories and symbolic links only, and is a small data set—3 cylinders is the default size.

The sysplex root provides access to all directories. Each system in a sysplex can access directories through the symbolic links that are provided. Essentially, the sysplex root provides redirection to the appropriate directories, and it should be kept very stable; updates and changes to the sysplex root should be made as infrequently as possible.

2. You need to define system-specific HFS data sets for each system in the sysplex that is using the sysplex HFS sharing capability. We recommend that you use a naming convention to associate this data set to the system for which it is defined. The sysplex root has symbolic links to mount points for system-specific etc, var, tmp, and dev HFS data sets.

   Note: The system-specific HFS data set should be mounted read/write. In addition, we recommend that the name of the system-specific data set contain the system name as one of the qualifiers. This allows you to use the &SYSNAME symbolic in BPXPRMxx.

   The mount point for these data sets will be dynamically created by the system during OMVS initialization.

   You can customize the BPXISYSS JCL in SYS1.SAMPLIB library to create a system-specific HFS. You need to run this job for each system in the sysplex that shares the sysplex root HFS. These data sets may be SMS- or non-SMS managed.

3. To set up an HFSplex, an OMVS CDS will need to be created. The CDS contains the sysplex-wide mount table and information about all participating systems, and all mounted file systems in the sysplex. To allocate and format a CDS, customize and submit the BPXISCDS sample job in SYS1.SAMPLIB. The job will create two CDSs: one is the primary and the other is a backup that is referred to as the alternate. In BPXISCDS, you also need to specify the number of mount records that are supported by the CDS. For more information refer to Chapter 18, “Shared HFS in a sysplex”, in *z/OS UNIX System Services,* GA22-7800. The CDS is used as follows:

   a. The first system that enters the sysplex with SYSPLEX(YES) initializes the OMVS CDS. The CDS controls shared HFS mounts and will eventually contain information about all systems participating in sysplex HFS sharing in this sysplex. This system processes its BPXPRMxx Parmlib member, including all its ROOT and MOUNT statement information. It is also the designated owner of the byte range lock manager for the participating group. The MOUNT and ROOT information are logged in the CDS so that other systems that eventually join the participating group can read data about systems that are already active in the HFSplex.

b. Subsequent systems joining the participating group will read the information in the CDS and will perform all mounts. Any new BPXPRMxx mounts are processed and logged in the CDS. Systems already in the participating group will then process the new mounts added to the CDS.

Once the OMVS CDS data set has been defined, update the COUPLExx Parmlib member to define the primary and alternate OMVS CDS to XCF.

4. The version HFS is the IBM-supplied root HFS data set containing files and executables for z/OS elements. To avoid confusion with the sysplex root HFS data set, the IBM supplied root HFS data set is called the “version HFS”. In a sysplex environment, there could be a number of version HFS data sets, each denoting a different release or service level of z/OS. We recommend that you mount the version HFS in read-only mode. The version HFS should never be updated by anything except SMP and should not contain anything other than what is delivered by IBM. If your root HFS is currently mounted read/write, refer to the section entitled “*Post-Installation Actions for Mounting the Root HFS in Read-Only Mode*” in *z/OS UNIX System Services,* GA22-7800.

   A GoldPlex can be set up to share the version HFS data sets on the shared sysres without needing to set up sysplex-wide HFS sharing as long as the version HFS is only mounted read-only on all systems.

   A GoldPlex may not include the sharing of the Master Catalog and therefore user catalogs may or may not be shared. So, you will need to consider how you want to manage the HFS data set catalog entries. If the HFS data sets are cataloged in the Master Catalog you will need to define a process to keep the Master Catalogs in sync across the sysplex. An alternative is to have the HFS data sets in a user catalog that is shared by all system within the sysplex.

5. If you are going to use the shared sysplex root HFS structure, you will need to adjust the HFS file structure on the incoming system. Adjustments would include preparation of the system-specific and version HFS data sets of the incoming system to reflect the naming standards of the target sysplex. This is a requirement if a shared BPXPRMxx will be used in the target sysplex.

   We recommend that the name of the system-specific data sets contain the system name as one of the qualifiers. This allows you to use the &SYSNAME symbolic in BPXPRMxx. We do *not* recommend using &SYSNAME as one of the qualifiers for the version HFS data set name. Appropriate names may be the name of the target zone, &SYSR1, or any other qualifier meaningful to the system programmer. For more information about version HFS naming standards, see **22.5.2, “HFS considerations” on page 383**.

   The BPXPRMxx Parmlib member is used to specify the various parameters to control the UNIX file system. This member also contains information to control the OMVS setup and processing. We recommend that you use two different members: one containing the file system information, and the other containing OMVS information. You can point the OMVS parameter of IEASYSxx Parmlib member at both BPXPRMxx members. You will find that migrating from one release to another is easier if you use this method.

   You can use a common BPXPRMxx member for all the systems in the sysplex. There are four parameters in BPXPRMxx that are relevant to HFS sharing in a sysplex: SYSPLEX, VERSION, ROOT, and MOUNT. The following are the details of each parameter:

   a. SYSPLEX

      You should specify SYSPLEX(YES) to indicate that you wish to use sysplex HFS sharing. This parameter tells the system at the time of IPL to take part in HFS sharing across the sysplex.

b. VERSION

This statement dynamically creates a mount point at the time of IPL to mount the version HFS file. The version HFS is the IBM-supplied root HFS data set. You should specify a VERSION parameter which identifies your z/OS system release level. It is a good idea to use the system residence volser in the version HFS name. Different z/OS systems in the sysplex can specify different VERSION parameters in their BPXPRMxx member to allow different releases or service levels of the root HFS. We recommend that you mount the version HFS in read-only mode. Just as you should never update a running sysres, similarly, you should never update a running version HFS, and using read only sharing provides better performance than using sysplex sharing. For specific actions you have to take before you can mount the version HFS in read-only mode, see the section entitled "*Post-Installation Actions for Mounting the Root HFS in Read-Only Mode*" in *z/OS UNIX System Services,* GA22-7800.

Note: We do not recommend using &SYSNAME as one of the qualifiers for the version HFS data set name. Appropriate names may be the name of the target zone, &SYSR1, or any other qualifier meaningful to the system programmer. For more information about version HFS naming standards, see **22.5.2, "HFS considerations" on page 383**.

c. ROOT

Specify the name of the sysplex root data set. This data set must be mounted read/write.

d. MOUNT

You should specify the mount information for all the HFS files required for your system. If you used &SYSNAME as one of the qualifiers when you defined your system-specific HFS data sets, then you can create a single BPXPRMxx member for all systems in your sysplex.

You can specify two new parameters for the MOUNT statements. The SYSNAME parameter specifies the name of the z/OS system in the sysplex that should own the HFS data set being mounted. The AUTOMOVE parameter determines if another system in the sysplex can take ownership of the HFS data set if the owning system goes down.

We recommend that you use a common BPXPRMxx member for all systems in the target sysplex. Example 8-3 and Example 8-4 on page 122 contain examples of the two BPXPRMxx members.

If you wish to have a particular system issue the mount for some system, subsystem, or application-related HFS data sets, you can use the SYSNAME parameter on the MOUNT statement in BPXPRMxx, or you can set up a system-specific BPXPRMxx member that only has those HFS data set mount parameters specified. This BPXPRMxx member suffix would need to be added to the OMVS=(xx,yy) parameter of the appropriate IEASYSxx Parmlib member. Note that regardless of where the file system is mounted, if it is mounted read/write (and therefore sharable using Sysplex HFS sharing), it will be accessible from *all* systems in the HFSplex.

Example 8-3 BPXPRM00 parameters

```
MAXPROCSYS(1000)
 MAXPROCUSER(50)
 MAXUIDS(50)
 MAXFILEPROC(200)
 MAXPTYS(256)
 CTRACE(CTIBPX00)
 FILESYSTYPE TYPE(UDS) ENTRYPOINT(BPXTUINT)
```

```
 NETWORK DOMAINNAME(AF_UNIX)
         DOMAINNUMBER(1)
         MAXSOCKETS(10000)
         TYPE(UDS)
 FILESYSTYPE TYPE(INET) ENTRYPOINT(EZBPFINI)
 NETWORK DOMAINNAME(AF_INET)
         DOMAINNUMBER(2)
         MAXSOCKETS(60000)
         TYPE(INET)
 MAXTHREADTASKS(50)
 MAXTHREADS(200)
 IPCMSGNIDS    (500)
 IPCMSGQBYTES  (262144)
 IPCMSGQMNUM   (10000)
 IPCSHMNIDS    (500)
 IPCSHMSPAGES  (262144)
 IPCSHMMPAGES  (256)
 IPCSHMNSEGS   (10)
 IPCSEMNIDS    (500)
 IPCSEMNSEMS   (25)
 IPCSEMNOPS    (25)
 MAXMMAPAREA(4096)
 MAXASSIZE(41943040)
 MAXCPUTIME(2147483647)
 MAXSHAREPAGES(131072)
 FORKCOPY(COW)
 SUPERUSER(BPXROOT)
 TTYGROUP(TTY)
 STARTUP_PROC(OMVS)
```

Example 8-4 BPXPRMFS - filesystem definitions

```
/*************************************************************/
/*  This member, as specified in IEASYS00, will cause        */
/*  OpenEdition to come up with all filesystems mounted.     */
/*************************************************************/
FILESYSTYPE TYPE(HFS)           /* Filesystem type HFS           */
            ENTRYPOINT(GFUAINIT) /* Entrypoint for defining HFS   */
            PARM(' ')           /* Null PARM for physical file   */
                                /* system                        */

VERSION('&SYSR1.')              /* version                       */
SYSPLEX(YES)                    /* sysplex enabled               */

FILESYSTYPE TYPE(TFS)           /* Type of file system to start  */
            ENTRYPOINT(BPXTFS)  /* Entry Point of load module    */
                                /* system                        */
FILESYSTYPE TYPE(AUTOMNT)       /* Type of file system to start  */
            ENTRYPOINT(BPXTAMD) /* Entry Point of load module    */
                                /*                               */
ROOT     FILESYSTEM('OMVS.SYSPLEX.ROOT')
                                /* OS/390 root filesystem        */
         TYPE(HFS)              /* Filesystem type HFS           */
         MODE(RDWR)             /* Mounted for read/write        */
                                /*                               */
MOUNT FILESYSTEM('OMVS.&SYSNAME..SYSTEM.HFS')
                                /* system-specific HFS           */
      MOUNTPOINT('/&SYSNAME.')
         TYPE(HFS)              /* Filesystem type HFS           */
         MODE(RDWR)             /* Mounted for read/write        */
```

```
         NOAUTOMOVE              /*                                */
                                 /*                                */
MOUNT FILESYSTEM('OMVS.&SYSR1..ROOT')
                                 /* Version HFS                    */
      MOUNTPOINT('/$VERSION')
         TYPE(HFS)               /* Filesystem type HFS            */
         MODE(READ)              /* Mounted for read/write         */
                                 /*                                */
MOUNT FILESYSTEM('OMVS.&SYSNAME..ETC')
                                 /* HFS for /etc directory         */
      MOUNTPOINT('/&SYSNAME./etc')
         TYPE(HFS)               /* Filesystem type HFS            */
         MODE(RDWR)              /* Mounted for read/write         */
         NOAUTOMOVE              /*                                */
                                 /*                                */
MOUNT FILESYSTEM('OMVS.&SYSNAME..VAR')
                                 /* HFS for /var directory         */
      MOUNTPOINT('/&SYSNAME./var')
         TYPE(HFS)               /* Filesystem type HFS            */
         MODE(RDWR)              /* Mounted for read/write         */
         NOAUTOMOVE              /*                                */
                                 /*                                */
MOUNT FILESYSTEM('LSK130.&SYSNAME..MSYS.HFS')
                                 /* HFS for msys for setup         */
      MOUNTPOINT('/lsk130')
         TYPE(HFS)               /* Filesystem type HFS            */
         MODE(RDWR)              /* Mounted for read/write         */
         NOAUTOMOVE              /*                                */
                                 /*                                */
MOUNT FILESYSTEM('LSK130.&SYSNAME..MSYS.LOG.HFS')
                                 /* HFS for msys for setup log file*/
      MOUNTPOINT('/lsk130/log')
         TYPE(HFS)               /* Filesystem type HFS            */
         MODE(RDWR)              /* Mounted for read/write         */
         NOAUTOMOVE              /*                                */
                                 /*                                */
MOUNT FILESYSTEM('/&SYSNAME./TMP') /* TFS for /tmp directory       */
      MOUNTPOINT('/&SYSNAME./tmp')
         TYPE(TFS)               /* Filesystem type TFS            */
         PARM('-s 10')           /* 10 meg file system             */
         NOAUTOMOVE              /*                                */
                                 /*                                */
                                 /*                                */
MOUNT FILESYSTEM('/&SYSNAME./DEV') /* TFS for /dev directory       */
      MOUNTPOINT('/&SYSNAME./dev')
         TYPE(TFS)               /* Filesystem type TFS            */
         PARM('-s 5')            /* 5 meg file system              */
         NOAUTOMOVE              /*                                */
                                 /*                                */
```

In z/OS 1.3 and later, there is a new MOUNT parameter—the UNMOUNT parameter specifies that the HFS is to be automatically unmounted when the owning system leaves the sysplex.

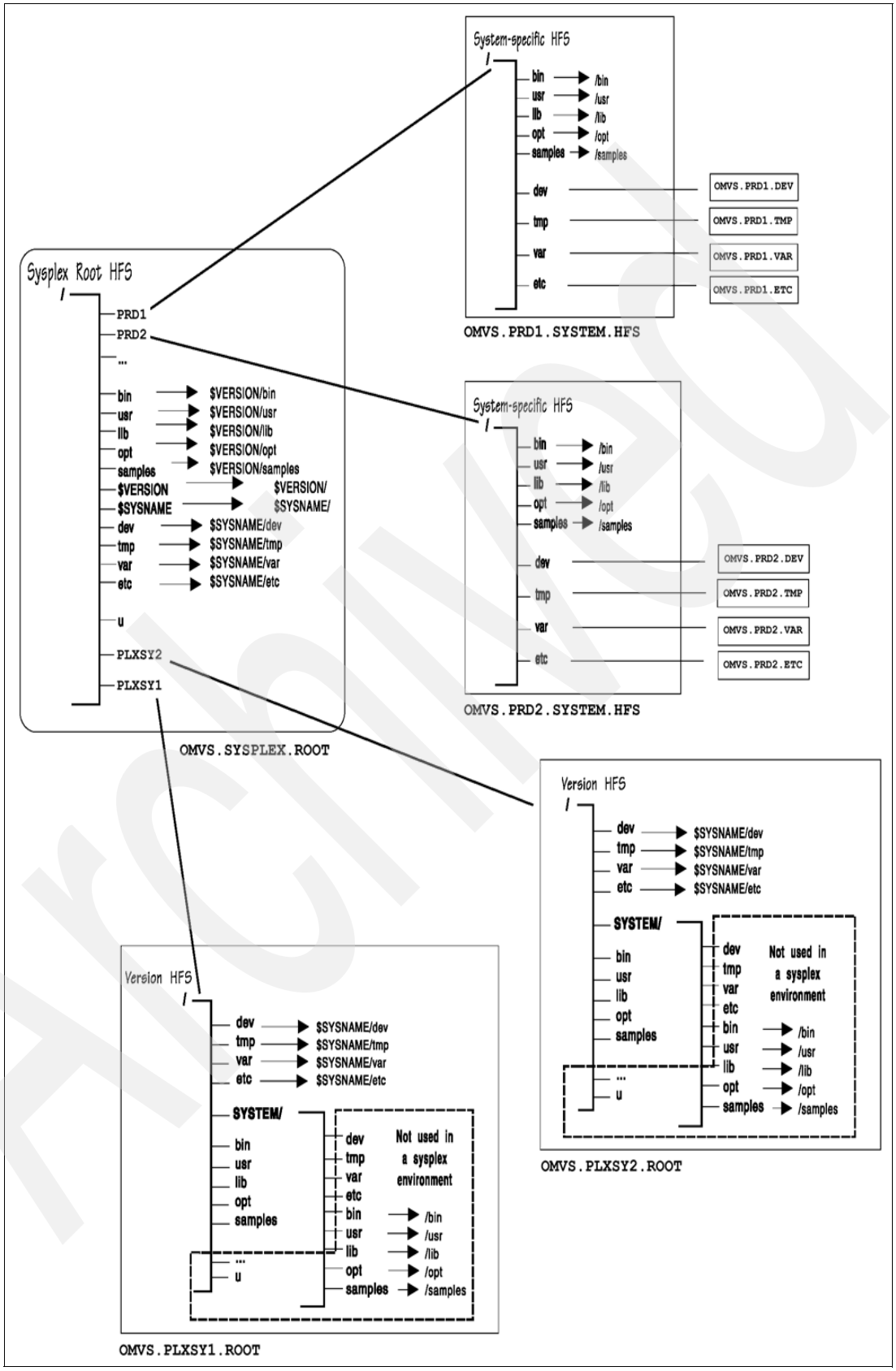


Figure 8-1 Shared HFS structure

In Figure 8-1 on page 124, if the content of the symbolic link begins with $VERSION or $SYSNAME, the symbolic link will resolve in the following manner:

- If you have specified SYSPLEX(YES) and the symbolic link for /dev has the contents $SYSNAME/dev, the symbolic link resolves to /PRD1/dev on system PRD1 and /PRD2/dev on system PRD2.
- If you have specified SYSPLEX(YES) and the content of the symbolic link begins with $VERSION, $VERSION resolves to the value nnnn specified on the VERSION parameter. Thus, if VERSION in BPXPRMxx is set to PLXSY1 (in our example BPXPRMxx we use &SYSR1 which substitutes the sysres id, in the case PLXSY1), then $VERSION resolves to /PLXSY1. For example, a symbolic link for /bin, which has the contents $VERSION/bin, resolves to /PLXSY1/bin on a system whose $VERSION value is set to PLXSY1.

6. UNIX System Services security (HFS)

   UNIX System Services (USS) is tightly integrated with the z/OS Security Server (RACF). The system programmer or data administrator must know the concepts of UNIX and USS security to manage the individual hierarchical file system (HFS) files (not data sets). It is especially important to understand the concept of a superuser. This topic provides some background information about UNIX and USS security. It also provides information about HFS security.

   a. UNIX file security overview

      On UNIX systems, each user needs an account that is made up of a user name and a password. UNIX uses the /etc/passwd file to keep track of user names and encrypted passwords for every user on the system (this file is not used in z/OS). Internally, the UNIX operating system uses a numeric ID to refer to a user, so in addition to user name and password, the /etc/passwd file also contains a numeric ID called the user identifier or UID. UNIX operating systems differ, but generally these UIDs are unsigned 16-bit numbers, ranging from zero to 65,535. z/OS UNIX System Services supports UID numbers up to 2,147,483,647.

   b. UNIX users and superuser

      There is no convention for assigning UID numbers to users other than 0 (zero), which has special significance in that it indicates a superuser who has special powers and privileges under UNIX. So, care must be exercised in deciding who can gain access to a UID of 0 (zero). If the z/OS Security Server (RACF) is installed, it offers some features that can be activated to gain more control over superusers.

      The z/OS Security Server also has capabilities to allow superuser-type access for some specific services for specific users. The UID is the actual number that the operating system uses to identify the user. User names are provided for convenience, as an easy way for us to remember our sign-on to the UNIX system. If two users are assigned the same UID, UNIX views them as the same user, even if they have different user names and passwords. Two users with the same UID can freely read and write each other files and can kill each other's processes.

      Note: If the users of the incoming system and target sysplex have the same UID, they will have access to the same resources and they will be able to cancel the same processes. During the merge of the systems, we recommend that you analyze users and UIDs according to the security policy of the target sysplex.

c. UNIX groups

UNIX systems also use the concept of groups, where you group together many users who need to access a set of common files, directories, or devices. Like user names and UIDs, groups have both group names and group identification numbers (GIDs). Each user belongs to a primary group that is stored in the /etc/passwd file on UNIX systems (this file is not used in z/OS USS). The z/OS UNIX System Services supports GID numbers up to 2,147,483,647.

d. Permission bits

All UNIX files have three types of permissions:

- Read (displayed as r)
- Write (displayed as w)
- Execute (displayed as x)

For every UNIX file, read, write and execute (rwx) permissions are maintained for three different types of file user:

- The file owner
- The group that owns the file
- All other users

The permission bits are stored as three octal numbers (3 bits for each type of file user) totalling nine bits. When displayed by commands, such as `ls -l`, a ten-character field is shown, consisting of nine for permission, preceded by one for file type.

Permission bit structure: The structure of the ten-character field is **tfffgggooo**, where:

t The type of file or directory. Valid values are:

- File

c Character special file

d Directory

l Symbolic link

p FIFO special file

fff The OWNER permissions, as explained in Table 8-2.

ggg The GROUP permissions, as explained in Table 8-2.

ooo The OTHER (or sometimes referred to as WORLD permissions), as explained in Table 8-2.

Table 8-2 File access type and permission bits

Pos	Char	Access Type	Permission for file	Permission for directory
1	r	read	Permission to read or print the contents.	Permission to read, but not search, the contents.
2	w	write	Permission to change, add to, or delete from the contents.	Permission to change, add, or delete directory entries.

Pos	Char	Access Type	Permission for file	Permission for directory
3	x	execute or search	Permission to run the file. This permission is used for executable files. For OWNER, the third position can also be displayed as: -**s** to indicate an executable file with set-user-ID set. -**S** to indicate a non-executable file with set-user-ID set. For GROUP, the third position can also be displayed as: -**s** to indicate an executable file with set-group-ID set. -**S** to indicate a non-executable file with set-group-ID set. For OTHER, the third position can also be displayed as: -**t** to indicate an executable file with sticky bit set. -**T** to indicate a non-executable file with sticky bit set.	Permission to search the directory.
any	-	No access		

Octal value bits: There are many places in UNIX where permission bits are also displayed as a representation of their octal value. When shown this way, a single digit is used to represent an rwx setting. The meaning associated with the single digit is:

- **0** No access (---)
- **1** Execute-only access (--x)
- **2** Write-only access (-w-)
- **3** Write and execute (-wx)
- **4** Read-only access (r--)
- **5** Read and execute access (r-x)
- **6** Read and write access (rw-)
- **7** Read, write, and execute access (rwx)

Permission bit examples: Remembering that each file always has permission bits for owner, group and other, it is usual to see three-digit numbers representing the permission bits of a file. For example, some typical file permission settings are:

666 `Owner('6'='rw-') group('6'='rw-') other('6'='rw-')`

700 `Owner('7'='rwx') group('0'='---') other('0'='---')`

755 `Owner('7'='rwx') group('5'='r-x') other('5'='r-x')`

777 `Owner('7'='rwx') group('7'='rwx') other('7'='rwx')`

User settings: A user may set permission bits for any combination at any level of access. For example, if a user wanted to have read, write, and execute access to one of his/her own files, but not allow access to anyone else, the permission bits would be set to 700, which the `ls -l` command would display as -rwx------.

z/OS 1.3 provides a new concept of Access Control Lists. For more information about this, see *z/OS Distributed File Servce zSeries File System Implementation,* SG24-6580 or *z/OS UNIX System Services,* GA22-7800.

e. UNIX file security with the z/OS Security Server (RACF)

With z/OS UNIX System Services, the concept of user accounts is the same as for any UNIX system, but the method of storing this account information is different. RACF, when used with z/OS UNIX System Services, integrates the UNIX account information with the existing MVS account and system information to provide a central secure database in which to store all security information. Each UNIX user must have a UID and GID assigned to them. These UIDs and GIDs are used by UNIX System Services to control or check access to files and processes.

UNIX System Services security functions are implemented in RACF partially as modifications to existing RACF functions, and partially as new RACF functions. The security functions provided include user validation, file access checking, and privileged user checking. UNIX System Services users are defined with RACF commands. When a job starts or a user logs on, the user ID and password are verified by existing MVS and RACF functions. When an address space requests a UNIX System Services function for the first time, RACF does the following:

i. Verifies that the user is a valid UNIX System Services user—that is, that the user has been assigned a UID.

ii. Verifies that the user's current connect group is a valid UNIX System Services group—that is, that the current connect group has been assigned a GID.

iii. Initializes the control blocks needed for subsequent security checks.

When the first initialization is done, file security checking is started. RACF does the following:

i. Checks whether a started task has trusted or privileged attributes. If a started task has those attributes, it is treated like superuser.

ii. Checks whether the user is superuser (uid=0).

iii. Checks whether the UID of the user is the UID of owner of the file. The owner permission is used by user. If the owner permission bit is 0, the user's file access is denied.

iv. Checks whether the GID of the user is the GID of owner of the file. The group permission is used by user. If the group permission bit is 0, the user's file access is denied.

v. Checks whether the other permission is not 0. If the other permission is 0, the user's file access is denied.

Note: RACF does not perform security-level checking for a started task that has the RACF privileged and trusted attribute. A started task with the privileged or trusted attribute is treated like a superuser (uid=0). This means that any started task with these attributes is able to access and/or cancel any PID.

f. RACF database

The RACF database contains, among other things, user profiles and group profiles. Associated with them is a new segment called the OMVS segment. For a user, the OMVS segment contains the user identifier (UID), initial directory path name (HOME), and initial program to execute after logon (PROGRAM). To display this information, the OMVS keyword should be appended to the `RACF LU TSO` command. Example 8-5 on page 129 shows how you would display the OMVS segment for user ALBERTO.

Example 8-5 Listuser command

```
LU ALBERTO OMVS NORACF

 USER=ALBERTO

 OMVS INFORMATION
 ----------------
 UID= 0000000314
 HOME= /u/alberto
 PROGRAM= /bin/sh
 CPUTIMEMAX= NONE
 ASSIZEMAX= NONE
 FILEPROCMAX= NONE
 PROCUSERMAX= NONE
 THREADSMAX= NONE
 MMAPAREAMAX= NONE
```

The NORACF keyword prevents the standard non-OMVS related information from being displayed.

For a group, the OMVS segment contains the group identifier (GID). To display this information, the OMVS keyword should be appended to the **RACF LG TSO** command. Example 8-6 shows how you would display the OMVS segment for group OMVSGRP.

Example 8-6 Listgrp command

```
LG OMVSGRP OMVS NORACF

INFORMATION FOR GROUP OMVSGRP

OMVS INFORMATION
----------------
GID= 0000000001
```

For more information, refer to *z/OS UNIX System Services,* GA22-7800 and *z/OS Security Server RACF Command Language Reference,* SA22-7687.

7. If you use SMS to manage your HFS data sets, you will need to review the ACS routines on the incoming system to ensure the HFS data sets are managed appropriately when merged into the target sysplex so the management of the data sets is consistent.

   This would be an issue primarily in a GoldPlex configuration because of the separate SMSplexes. In a PlatinumPlex configuration there would be a single SMSplex, so this would not be an issue.

8. The following considerations apply when using zFS in a sysplex in shared HFS mode:
   - Only systems running the zFS address space can use zFS file systems. The file system hierarchy appears different when viewed from systems with zFS-mounted file systems than it does from those systems not running zFS. Pathname traversal through zFS mountpoints will have different results in these cases since the zFS file system is not mounted on those systems not running zFS.
   - zFS file systems owned by another system are accessible from a member of the sysplex that is running zFS.
   - zFS compatibility mode file systems can be automoved and automounted. A zFS compatibility mode file system can only be automoved to a system where zFS is running.

- You can have multi-file system aggregates in sysplex sharing mode, but they should be mounted NOAUTOMOVE (or with UNMOUNT, if running z/OS 1.3 or later) to automatically unmount the file systems when the owning system leaves the sysplex.
- The *IOEFSPRM* file cannot be shared across systems in a sysplex when the file contains:
  - A multi-file system aggregate specification, or
  - A *msg_output_dsn* specification, or
  - A *trace_dsn* specification.

  In addition, you cannot share an *IOEFSPRM* file that has a *user_cache_size* specification of greater than 2048 M if there are systems that are running at a level lower than z/OS 1.3. In this case, you should use the *&SYSNAME* system variable in the *IOEZPRM DD* of the ZFS PROC to specify different *IOEFSPRM* files for different systems.

  In z/OS V1.4, system variables can be used, so you can specify system-specific multi-file system aggregates and other data sets by using &SYSNAME, for example.

  If you are only using compatibility mode aggregates and file systems, and you are not specifying a *msg_output_dsn* or a *trace_dsn*, and you use the same options for all ZFS PFSs on all systems, you can share the same *IOEFSPRM* file across systems.

9. The following describes z/OS UNIX considerations that relate to the level of z/OS running on the members of the sysplex:
   - When all members of the sysplex are at z/OS 1.2 and some, or all, systems are running zFS:
     - All systems running zFS see the zFS file systems. The file system hierarchy appears differently when viewed from systems with zFS mounted file systems than it does from those systems not running zFS. Pathname traversal through zFS mountpoints will have different results in those cases, since the zFS file system is not mounted on those systems not running zFS.
   - If a system running zFS is brought down:
     - zFS compatibility mode file systems owned by that system that are allowed to be automoved will be automoved to another system running zFS. If this function fails to find another owner, the file system becomes unowned.
     - zFS file systems that are specified as NOAUTOMOVE become unowned. To be sure that a file system gets unmounted, if you are not using the UNMOUNT command introduced in z/OS V1.3, you should do an explicit UNMOUNT before taking down the system.
     - File systems which are unowned are not visible in the file system hierarchy, but can be seen from a **`D OMVS,F`** operator command. To recover a file system that is mounted and unowned, the file system must be unmounted.
   - If zFS is brought down on one system in the sysplex:
     - zFS compatibility mode file systems owned by the system that are allowed to be automoved will be automoved to another system running zFS. If this function fails to find another owner, the file system and all file systems mounted under it are unmounted in the sysplex.
     - zFS file systems that are defined as NOAUTOMOVE and all file systems mounted under them are unmounted in the sysplex.

– When all members of the sysplex are *not* at z/OS 1.2 and some or all systems are running zFS:
  - Only systems running zFS see zFS file systems. The file system hierarchy appears differently when viewed from systems with zFS mounted file systems than it does from those systems not running zFS. Pathname traversal through zFS mountpoints have different results in such cases since the zFS file system is not mounted on those systems not running zFS.
– If a system running zFS is brought down:
  - zFS compatibility mode file systems owned by the system that can be automoved are automoved to another system running zFS. If this function fails to find another owner, the file system becomes unowned.
  - zFS file systems owned by the system that are noautomove become unowned.
  - File systems which are unowned are not visible in the file system hierarchy, but can be seen from a **`D OMVS,F`** operator command. To recover a file system that is mounted and unowned, the file system must be unmounted.
– If zFS is brought down on a system:
  - All zFS file systems owned by any of the systems unmount, even if this zFS does not own any zFS file system.

> **Note:** This means that terminating zFS on any one system causes all zFS file systems in the sysplex to be unmounted. Because of this, it is not recommended at this time to allow zFS file systems to be shared between systems in a shared HFS environment when all systems are not at z/OS 1.2 or higher. In a sysplex environment where all members are not running z/OS 1.2 or higher, the following configuration choices are recommended:
>
> ► Restrict zFS usage to one member of the shared HFS sysplex group, or,
>
> ► Restrict zFS usage to images not participating in a shared HFS sysplex group.
>
> Alternatively, you should restrict zFS usage to a multi-system shared HFS sysplex where all systems are at z/OS 1.2 or higher.

8.3 Methodology for merging HFSplexes

In this section, we discuss an approach that can be used to either merge or extract data from an HFS. Before you merge the incoming system into the target sysplex, you must “clean up” the root HFS to move all the non-read only data out of the root. To do this, you create system-specific HFS data sets for the /etc and /var directories, move the contents of those directories out of the root HFS into those new data sets, and switch over the new HFS data sets. This positions you to easily move the incoming system into the target sysplex HFSplex when you bring it into the sysplex.

In the examples provided, we outline the process that could be used to extract the required directories from the version root HFS into the system-specific HFS data sets. We use the system name of “TEST” as the incoming system name.

The steps required are:

1. Define the required system-specific HFS data sets. These are the target HFS data sets that will contain the contents of the /etc and /var directories that are copied from the version HFS.

   Example 8-7 contains sample JCL to do this:

Example 8-7 Define system specific data sets

```
//DEFHFS   JOB ,'DEFINE',MSGCLASS=A,MSGLEVEL=(1,1)
//*
//STEP01 EXEC  PGM=IEFBR14
//DD1    DD   DSNAME=OMVS.TEST.ETC,
//            SPACE=(TRK,(40,1,1)),
//            DSNTYPE=HFS,
//            DISP=(NEW,CATLG)
//DD1    DD   DSNAME=OMVS.TEST.VAR,
//            SPACE=(TRK,(40,1,1)),
//            DSNTYPE=HFS,
//            DISP=(NEW,CATLG)
```

2. Ensure the userid you are using has the appropriate access to perform the required tasks. For example, you will require superuser access to issue the mounts. If not, get access to the BPX.SUPERUSER FACILITY class.
3. Create temporary mount points and mount the system-specific HFS data sets at /tmp directory mount points. This is required to make the newly-created HFS data sets available to copy the required data into.

   Example 8-8 contains a sample REXX to do this:

Example 8-8 Create /tmp directories and mount required HFS

```
/* REXX */
parse source . . . . . . . omvs .
tso= omvs<>'OMVS'
if tso then call syscalls on
   call syscall 'mkdir /tmp/var 755'
   call syscall 'mkdir /tmp/etc 755'
address tso
   "MOUNT FILESYSTEM('OMVS.TEST.VAR') ",
   "MOUNTPOINT('/tmp/var') TYPE(HFS)"
   "MOUNT FILESYSTEM('OMVS.TEST.ETC') ",
   "MOUNTPOINT('/tmp/etc') TYPE(HFS)"
exit
syscall:
   parse arg cmd
   address syscall cmd
   return
```

4. Prior to copying the /etc and /var data from the version HFS, to guarantee integrity, you will need to ensure that these directories are not being updated while the data is copied. You may need to do the copy step during a change window or quiet period. To ensure there are no updates during the copy process, you should mount the version HFS as READ only.

   You can do this a number of ways, via ISHELL, OMVS or TSO. For example, in TSO you could issue the following command, using the appropriate root HFS data set name:

Example 8-9 Mount version HFS as read/only prior to copy

```
TSO UNMOUNT FILESYSTEM('OMVS.TEST.ROOT') REMOUNT(READ)
```

5. The next step is to copy the /etc and /var directories from the version HFS into their own system-specific HFS data sets. There are a number ways to do this, including using the “pax” or “tar” UNIX commands. These commands have numerous and complex command parameters that are required to ensure the file and directory structure and the file attributes are preserved during the copy.

   To avoid this complexity, we recommend that you obtain a copy of the COPYTREE utility. COPYTREE is a freely available utility that can run under TSO or the shell and is used to make a copy of a file hierarchy preserving all file attributes. For information about COPYTREE utility, refer to 8.4, “Tools and documentation” on page 134.

   Issue COPYTREE commands to copy the directory contents from the version HFS into the system-specific /etc and /var HFSs. Example 8-10 contains a sample REXX to do this:

Example 8-10 Copy directory contents using COPYTREE

```
/* REXX */
parse source . . . . . . . omvs .
tso= omvs<>'OMVS'
if tso then call syscalls on
address tso
"COPYTREE /etc /tmp/etc"
"COPYTREE /u /tmp/var"
exit
```

6. You now have the contents of the /etc and /var directories in their own HFS data sets. You now need to mount these HFS data sets at the appropriate mount points to ensure these are used, instead of the version HFS data set. To do this, you will need to unmount the /etc and /var HFS data sets from the /tmp mountpoint and remount them at the /etc and /var mountpoints. Example 8-11 contains a sample REXX to do this:

Example 8-11 Mount the new /etc and /var HFS data sets

```
/* REXX */
parse source . . . . . . . omvs .
tso= omvs<>'OMVS'
address tso
   "UNMOUNT FILESYSTEM('OMVS.TEST.VAR') NORMAL"
   "UNMOUNT FILESYSTEM('OMVS.TEST.ETC') NORMAL"
   "MOUNT FILESYSTEM('OMVS.TEST.VAR') ",
   "MOUNTPOINT('/var') TYPE(HFS) MODE(RDWR)"
   "MOUNT FILESYSTEM('OMVS.TEST.ETC') ",
   "MOUNTPOINT('/etc') TYPE(HFS) MODE(RDWR)"
exit
```

7. With the new /etc and /var HFS data sets mounted and in use, you can remount the version HFS read/write, if required.

   You can do this a number of ways, via ISHELL, OMVS or TSO. For example, in TSO you could issue the following command, using the appropriate root HFS data set name:

Example 8-12 Mount version HFS RDWR after the copy

```
TSO UNMOUNT FILESYSTEM('OMVS.TEST.ROOT') REMOUNT(RDWR)
```

8. The last step is to update the BPXPRMxx member of Parmlib to ensure the new created /etc and /var HFS data sets are mounted at the /etc and /var mountpoints at the next IPL.

8.4 Tools and documentation

In this section we provide information about tools and documentation that may help you complete this task.

8.4.1 Tools

The following tools should be used to help you successfully complete the merge:

- BPXISYSR sample job in SYS1.SAMPLIB creates the sysplex root.
- BPXISYSS sample job in SYS1.SAMPLIB creates the system-specific HFS data sets.
- COPYTREE is a utility that can run under TSO or the shell and is used to make a copy of a file hierarchy, preserving all file attributes. It can also be used to check the integrity of file system structures without copying the file tree. It is available from IBM's z/OS UNIX Tools and Toys Web site at URL:

  `http://www.ibm.com/servers/eserver/zseries/zos/unix/bpxalty2.html`

8.4.2 Documentation

In addition, the following documentation contains information that will help you during this exercise:

- *Hierarchical File System Usage Guide,* SG24-5482
- *UNIX System Services File System Interface Reference,* SA22-7808
- *z/OS Distributed File Servce zSeries File System Implementation,* SG24-6580
- *z/OS Security Server RACF Command Language Reference,* SA22-7687
- *z/OS UNIX System Services,* GA22-7800

9

Language Environment considerations

This chapter discusses the considerations for Language Environment (LE) when merging a system into an existing sysplex.

9.1 Language Environment

To fully understand the considerations for Language Environment (LE) when merging the incoming system into the target sysplex, you need to understand which option(s) are used to access the LE run-time libraries in both environments.

This chapter does not discuss LE conversion of applications, nor is it a requirement of the subsystems (for example, CICS) to convert to use LE as part of this process. The chapter discusses the LE run-time environment with the view to ensuring the elements and features of z/OS and related subsystems that are using LE continue to function as required on the incoming system and within the target sysplex.

Language Environment provides a common run-time environment for IBM versions of certain high-level languages (HLLs), namely C, C++, COBOL, Fortran, and PL/I, in which you can run existing applications written in previous versions of these languages as well as in the current Language Environment-conforming versions. Prior to Language Environment, each of the HLLs had to provide a separate run-time environment. Language Environment provides compatible support for existing HLL applications; most existing single-language applications can run under Language Environment without being recompiled or relink-edited.

Applications that require the run-time library provided by Language Environment can access the run-time data sets (SCEERUN and SCEERUN2) using LNKLST or STEPLIB - or a combination of the two methods:

- LNKLST

  The Language Environment run-time libraries, SCEERUN and SCEERUN2, can be placed in LNKLST and heavily used modules can be placed in LPA, if required.

- STEPLIB

  If the SCEERUN and SCEERUN2 data sets cannot be placed in LNKLST, you can STEPLIB the data sets for each application that requires them.

There are a number of elements and features of the operating system that require LE to function. With each release of OS/390, and now z/OS, more and more elements and features come to rely on LE to function correctly. The easiest way of ensuring these functions and elements of z/OS work correctly is to place the LE run-time libraries, SCEERUN and SCEERUN2, ahead of any other language run-time libraries in the LNKLST concatenation (and we recommend that Language Environment be the only run-time libraries in LNKLST). However, when adding Language Environment run-time libraries to LNKLIST is not possible, STEPLIB may be used, see the z/OS program directory for details.

You will need to ensure that applications are not adversely affected when merging the incoming system into the target sysplex by reviewing the installation-wide Language Environment run-time options prior to the merge. The Language Environment run-time option values control such features as:

- The national language in which messages appear
- How a debug tool is invoked
- When condition handling is invoked
- How storage is allocated to the heap and stack
- Shared storage allocations

LE has two default option modules: CEECOPT for CICS, and CEEDOPT for non-CICS, such as batch programs. They provide installation-wide defaults. The installation defaults distributed by IBM can be changed by creating a new CEECOPT or CEEDOPT module and installing it with an SMP/E usermod.

LE provides a facility to enable a specific application to override the default run-time option values, using the user options module, CEEUOPT. The CEEUOPT module is linked with the application program.

LE also provides a region-wide default options module, CEEROPT. CEEROPT addresses the requirement to have different run-time options for individual CICS regions. It resides in a user-specified load library. CEEROPT can be used with CICS and IMS with Library Routine Retention (LRR). If you are running programs that require Language Environment in an IMS/TM dependent region, such as an IMS message processing region, you can improve performance if you use Language Environment library routine retention. For more information see Chapter 9, “Using Language Environment under IMS” in *Language Environment for OS/390 Customization*, SC28-1941 or *z/OS Language Environment Customization*, SA22-7564.

BronzePlex considerations

In line with the sysplex target environment options outlined in 1.2, “Starting and ending points” on page 2, moving the incoming system into a BronzePlex would typically be done to obtain the benefits of PSLC or WLC. This would mean that the incoming system would not be sharing the sysres or maintenance environment. Therefore, there should not be any LE considerations in relation to the BronzePlex implementation.

GoldPlex considerations

Moving the incoming system into a GoldPlex would be done to share the same system software environment (sysres) and maintenance environment, and therefore you will need to review this chapter to ensure any LE considerations are addressed. The main consideration with merging into a shared sysres environment is the potential for the run-time options being different between the systems.

PlatinumPlex considerations

Moving the incoming system into a PlatinumPlex would be for system level resource sharing within the sysplex, which includes a shared sysres, maintenance environment, Master Catalog and all user catalogs, and so on. This configuration would allow you to obtain the maximum benefits from being in the sysplex. For example, the PlatinumPlex configuration would allow you to use ARM for subsystem restarts on the incoming system, and potentially use the incoming system as an additional target for batch workloads from the systems in the target sysplex. The main LE consideration with merging into a shared sysres environment is the potential for the run-time options being different between the systems.

9.2 LE considerations when merging systems into a sysplex

When moving the incoming system into the target sysplex, the number of LE considerations will depend on the functions and features you plan to exploit within the sysplex. To help understand the requirements, review the following considerations:

Table 9-1 LE considerations when merging systems into a sysplex

Consideration	Note	Type	Done?
Review considerations if the incoming system is going to share the sysres with other systems in the target sysplex	**1**	**G, P**	
Review considerations if the incoming system going to be used as a target for subsystem restarts using ARM or similar automatic restart mechanism	**2, 3**	**P**	

Consideration	Note	Type	Done?
Review considerations if the incoming system going to be used as a resource for sysplex batch workloads within the target sysplex	**2, 3**	**P**	
Handling if either the incoming system or the target sysplex will be used by application development to compile programs	**4**	**G, P**	

The “Type” specified in Table 9-1 on page 137 relates to the sysplex target environment—**B** represents a BronzePlex, **G** represents a GoldPlex, and **P** represents a PlatinumPlex. If none of these considerations apply (in other words, if the target environment is a BronzePlex), then there should not be any LE considerations when moving the incoming system into the target sysplex.

Notes in Table 9-1 on page 137 are described below:

1. You will need to review LE run-time options if the incoming system is going to share the sysres with other systems within the target sysplex. Ensure the run-time options being used on the target sysplex are compatible with those on the incoming system prior to the implementation. Refer to 9.4, “LE run-time options” on page 139.
2. You will need to consider downward compatibility issues:
   - If the incoming system is to be used as a target for subsystem restarts using ARM or a similar automatic restart mechanism, and the incoming system is at a lower operating system level than the target sysplex.
   - If the incoming system is to be used as a resource for the sysplex batch workload within the target sysplex, and the incoming system is at a lower operating system level than the target sysplex.

   Review 9.3.2, “Downward compatibility” on page 139.

   We don’t recommend these configurations when merging a system into a sysplex. We *do* recommend first bringing the incoming system up to the same operating system and maintenance level as the target sysplex. This may, however, be a valid configuration during a system upgrade cycle, after the merge.
3. You will also need to review the LE run-time options:
   - If the incoming system is to be used as a target for subsystem restarts using ARM, or a similar automatic restart mechanism, and the incoming system is *not* sharing the sysres with other systems within the target sysplex.
   - If the incoming system is to be used as a resource for the sysplex batch workload within the target sysplex, and the incoming system is *not* sharing the sysres with other systems within the target sysplex.

   You will need to review the LE run-time options to ensure the options being used on the incoming system are compatible with those on the target sysplex. We recommend that you also review the default settings, as they can change between releases. This will ensure that any applications running in a subsystem restarted on the incoming system (once it is a member of the target sysplex) will work as expected. Refer to 9.4, “LE run-time options” on page 139.

   We don’t recommend these configurations when merging a system into a sysplex. We *do* recommend sharing the sysres. This may however, be a valid configuration during a system upgrade cycle, after the merge.
4. You will also need to review your language(s) compile options if either the incoming system or the target sysplex is used by application development to compile programs, and

the incoming system is going to share the sysres with other systems within the target sysplex. Although this chapter is on LE, we thought it would be worth pointing this out.

9.3 Compatibility

Note the following compatibility support.

9.3.1 Upward compatibility

Language Environment is upwardly compatible. This means that applications coded and link-edited with one release of Language Environment will continue to run on later releases of Language Environment without the need to recompile or re-link edit the application.

9.3.2 Downward compatibility

As of OS/390 2.10, Language Environment provides downward compatibility support. Assuming that the required programming guidelines and restrictions are met, this support enables you to develop applications on higher release levels of OS/390 or z/OS for use on platforms that are running lower release levels of OS/390 or z/OS. For example, a company may use OS/390 2.10 with Language Environment on a development system where applications are coded, link-edited, and tested, while using any supported *lower* release of Language Environment on their production systems where the finished application modules are used.

Downward compatibility support is not the roll-back of new function to prior releases of OS/390 or z/OS. Applications developed that exploit the downward compatibility support must not use any Language Environment function that is unavailable on the lower release of OS/390 or z/OS where the application will be used. The downward compatibility support includes toleration PTFs for lower releases of OS/390 or z/OS to assist in diagnosing applications that do not meet the programming requirements for this support (specific PTF numbers can be found in the PSP buckets). The downward compatibility support provided by OS/390 2.10 (and above) and by the toleration PTFs does not change Language Environment's upward compatibility. Refer to the “Downward Compatibility Considerations” section in *Language Environment for OS/390 & VM Programming Guide*, SC28-1939, or *z/OS Language Environment Programming Guide*, SA22-7561.

9.4 LE run-time options

Ensure the run-time options being used on the target sysplex are compatible with those on the incoming system prior to implementation. Where differences are identified, you will need to make an informed decision on which option should be used, and understand the impact the change may have on the application. We recommend the decision be made in consultation with the appropriate subsystem and application development teams.

The LE run-time option defaults can vary between z/OS levels, and new option values are often added in a new release. We recommend you review the run-time options on the incoming system and the target sysplex. See Chapter 4, “Customizing Language Environment Run-Time Options” in *Language Environment for OS/390 Customization*, SC28-1941or *z/OS Language Environment Customization*, SA22-7564 for detailed information on the LE run-time options.

LE has two default option modules: CEECOPT for CICS, and CEEDOPT for non-CICS, such as batch programs. They provide installation-wide defaults. The installation defaults distributed by IBM can be changed by creating a new CEECOPT or CEEDOPT module and installing it with an SMP/E usermod, and it is these usermods that you will need to review to ensure compatibility. We recommend that you also compare the source of these usermods to the values from the running systems. These run-time option values can be obtained in batch for CEEDOPT (refer to 9.4.1, “Displaying the run-time values in batch” on page 140) and within CICS for CEECOPT (refer to 9.4.2, “Obtaining the run-time values in CICS” on page 142).

LE provides a facility to enable a specific application to override the default run-time option values, using the user options module, CEEUOPT. The CEEUOPT module is linked with the application program. When creating a CEEUOPT, only the run-time options and suboptions whose values are to be changed need to be specified. The application specifies the required run-time options, so merging the incoming system into the target sysplex should have no impact on the application.

LE also provides a region-wide default options module, CEEROPT. CEEROPT addresses the requirement to have different run-time options for individual CICS regions. It resides in a user-specified load library. CEEROPT can be used with CICS and IMS with LRR (Library Routine Retention). When creating a CEEROPT module, only the options whose values are to be changed need to be specified. The region-wide default options module resides in a user-specified load library which would generally be region(s) specific and not on a sysres, so merging the incoming system into the target sysplex should have no impact, but we recommend you identify where this module resides, if this option is being used.

9.4.1 Displaying the run-time values in batch

One way to obtain a listing of the values specified for the run-time options is by using the “options report” produced by the RPTOPTS(ON) option. For a batch program, this option can be specified on the PARM= field of the execute card. The general format for specifying run-time options in the PARM= field is:

```
//stepname EXEC PGM=program_name,
//         PARM='run-time options/program parameters'
```

For example:

Example 9-1 Find the current LE run-time options using a PL/I program

```
//STEP010 EXEC PGM=pliprog,PARM='RPTOPTS(ON)/'
```

For COBOL users with CBLOPTS(ON) as the installation default for the CBLOPTS run-time option, the slash should go *before* the LE run-time options:

```
//stepname EXEC PGM=program_name,
//         PARM='program parameters/run-time options'
```

For example:

Example 9-2 Find the current LE run-time options using a COBOL program

```
//STEP010 EXEC PGM=cobolpgm,PARM='/RPTOPTS(ON)'
```

Tip: Make sure you use SCEERUN as the run-time library when running your program.

This will produce a run-time options report similar to the one shown in Figure 9-1

```
Options Report for Enclave SAMPLE 04/25/02 3:26:07 PM
Language Environment V01 R02.00

LAST WHERE SET                OPTION
------------------------------------------------------------------------------
Installation default            ABPERC(NONE)
Installation default            ABTERMENC(ABEND)
Installation default          NOAIXBLD
Installation default            ALL31(OFF)
Installation default            ANYHEAP(16384,8192,ANYWHERE,FREE)
Installation default          NOAUTOTASK
... ... ...
Installation default            ERRCOUNT(0)
Installation default            ERRUNIT(6)
Installation default            FILEHIST
Default setting               NOFLOW
Installation default            HEAP(32768,32768,ANYWHERE,KEEP,8192,4096)
... ... ...
Installation default            INFOMSGFILTER(OFF,,,,)
Installation default            INQPCOPN
Programmer default              INTERRUPT(ON)
Installation default            LIBRARY(SYSCEE)
Installation default            LIBSTACK(4096,4096,FREE)
Installation default            MSGFILE(SYSOUT,FBA,121,0,NOENQ)
... ... ...
Installation default            RDRUNIT(5)
Installation default            RECPAD(OFF)
Invocation command              RPTOPTS(ON)
Installation default            RPTSTG(OFF)
Installation default          NORTEREUS
Installation default            RTLS(OFF)
Installation default          NOSIMVRD
... ... ...
```

Figure 9-1 Options report produced by LE run-time option RPTOPTS(ON)

In the run-time options report, the LAST WHERE SET column indicates where each option obtained its setting. In this column, look for the words `Installation default`. These options obtained their value from the installation default values (CEEDOPT/CEECOPT).

The run-time options which cannot be specified in CEEDOPT, CEEROPT, or CEEUOPT will be indicated by the phrase `Default setting`. If other phrases appear, this means the value for these run-time options is coming from other than the installation defaults as outlined in Table 9-2.

Table 9-2 LE run-time option report

Programmer default	The option was specified in CEEUOPT, C #pragma runopts, or PL/I PLIXOPT
Override	Indicates LE forced the setting of the option
Invocation command	Option was specified on the JCL in PARM= (you should see this for RPTOPTS)

9.4.2 Obtaining the run-time values in CICS

Under CICS, run-time options cannot be passed as parameters when the application is invoked. The RPTOPTS(ON) run-time option can be specified using one of the following methods:

- CEEROPT, which provides region-wide default options
- CEEUOPT, which is link-edited with the program to provide application specific defaults

Information on creating a CEEROPT or CEEUOPT CSECT is available in *Language Environment for OS/390 Customization*, SC28-1941 *z/OS Language Environment Customization*, SA22-7564.

If your installation is using LE with CICS, APAR PQ38838 provides support for the LE CICS CLER transaction. The CLER transaction will display the current run-time options for a CICS region, and also modify a subset of the options if required.

9.5 Tools and documentation

In this section we provide information about tools and documentation that may help you complete this task.

9.5.1 Tools

There are no tools to help with this activity.

9.5.2 Documentation

The following publications provide information that may be helpful in managing Language Environment:

- *Language Environment for OS/390 & VM Programming Guide*, SC28-1939
- *Language Environment for OS/390 Customization*, SC28-1941
- *z/OS Language Environment Programming Guide*, SA22-7561
- *z/OS Language Environment Customization*, SA22-7564

10

System data set considerations

This chapter discusses the considerations for merging and sharing system data sets when moving a system into an existing sysplex.

Here, we address the following questions:

- Is it necessary to merge sysres, Master Catalog, Parmlib and Proclib? Or is it possible to have more than one of them?

We also produce the following items:

- A checklist of things to consider should you decide to merge them
- A methodology for doing the merge
- A list of tools and documentation that can assist with this exercise

10.1 Introduction

When starting to implement a sysplex, an area that many installations look at first is the sharing of system data sets. The level of sharing will vary by installation, and depends on where you want to end up. In this chapter, we discuss the considerations for the following three target environments:

- The BronzePlex scenario is where the minimum number of resources are shared—this is basically a non-shared environment (several copies of each of the system data sets) with some special considerations because the systems are in a sysplex. Each copy must be maintained separately, with each one likely to be unique. System programmers must keep track of the differences, the reasons for those differences, and updates of the appropriate ones. When an update is introduced across a number of systems, multiple copies of the same data sets must be changed. Much of this process can be simplified by consolidating onto one or two sets of system data sets (e.g., Parmlib) per Parallel Sysplex cluster, and that is why the PlatinumPlex scenario is often more attractive.
- In a GoldPlex scenario, a subset of the system data sets is shared. For example, the sysres is shared among systems, but the Proclib data set might not be, especially if the incoming system is totally unrelated to the existing systems. For example, if you have just one JES2 started task JCL, it has to contain all the user Proclibs—however, half the volumes will be offline to each system, meaning that half the JES2 procs will be missing, so you would get a JCL error when you tried to start JES2.
- We have called the ideal scenario a PlatinumPlex. Instead of having multiple copies of system data sets, such as Parmlib, Proclib, Master Catalog, and sysres volumes, ideally you would have just one of each in a Parallel Sysplex. The consolidation and standardization of these resources can significantly ease the management of a z/OS environment. System programmers can maintain these libraries much more easily. For example, through the use of symbols and cloning, a change to one member of a data set can be sufficient to implement a change across many systems. This consolidation also leads to standardization of these system resources across the Parallel Sysplex, making it easier to implement changes and manage the systems.

  To make it possible to implement a PlatinumPlex, z/OS provides a number of capabilities such as system symbols, shared sysres, shared Master Catalog, and others. These are discussed in the following sections.

10.2 System symbols

System symbols are variables used to make system definitions more flexible in a shared environment. They are elements that allow systems to share definitions, while retaining unique values in those definitions. Each system that shares the definition replaces the system symbol with a unique value during initialization. They are set in Parmlib member IEASYMxx. For a comprehensive description of statements and parameters associated with system symbols, refer to the *z/OS MVS Initialization and Tunning Reference,* SA22-7592.

The following terms describe the elements of system symbols:

Symbol Name The name that is assigned to a symbol. It begins with an ampersand (&) and optionally ends with a period (.).

Substitution Text The character string that the system substitutes for a symbol each time it appears. Substitution text can identify characteristics of resources, such as the system on which a resource is located, or the date and time of processing. When you define static system symbols in the IEASYMxx Parmlib member (see “Step 5. Create an IEASYMxx

Parmlib Member" in *z/OS MVS Initialization and Tunning Reference,* SA22-7592), the substitution text can contain other static system symbols; the resolved substitution text refers to the character string that is produced after all symbols in the substitution text are resolved.

The following terms describe the types of system symbols:

Dynamic System Symbol

A system symbol whose substitution text can change at any point between IPLs. Dynamic system symbols represent values that can change often, such as dates and times. A set of dynamic system symbols is defined to the system; your installation cannot provide additional dynamic system symbols.

Static System Symbol A symbol whose substitution text is defined at system initialization and remains fixed for the life of an IPL. (One exception, &SYSPLEX, has a substitution text that can change at one point in an IPL; see "Step 6. Code Support for System Symbols in LOADxx" on page 46 of *z/OS MVS Initialization and Tunning Reference,* SA22-7592). Static system symbols are used to represent fixed values such as system names and sysplex names. Static system symbols have two types:

- *System-defined static system symbols* already have their names defined to the system. Your installation defines substitution texts or accepts system default texts for the static system symbols, which are:

  &SYSCLONE The last two characters of &SYSNAME system symbol

  &SYSNAME The name of the system

  &SYSPLEX The name of the sysplex

  &SYSR1 The IPL volume serial name **Note**: Your installation cannot define substitution texts for &SYSR1.

- *Installation-defined static system symbols* are defined by your installation. The system programmer specifies their names and substitution texts in the IEASYMxx member of the Parmlib concatenation.

System symbols can be used in most of the Parmlib members, system commands, JCL for started tasks and TSO/E logon procedures, JES2 initialization statements and commands, JES3 commands, dynamic allocations, and other places. The following are some examples of how system symbols can be used in a Parallel Sysplex consisting of two systems named SYSA and SYSB:

- Data set names: If the systems use the same SMFPRMxx Parmlib member, you can specify the following naming pattern to create different SMF data sets on each system: SYS1.&SYSNAME..MAN1. This definition produces:
  - SYS1.SYSA.MAN1 on system SYSA
  - SYS1.SYSB.MAN1 on system SYSB
- Parmlib members: If the systems use the same IEASYSxx Parmlib member and require different CONSOLxx Parmlib members, you can specify CON=&SYSCLONE. This definition results in:
  - CON=SA on system SYSA - so member CONSOLSA would be used on system SYSA

- CON=SB on system SYSB - so member CONSOLSB would be used on system SYSB

- Started task JCL: If JCL is for a started task, you can specify system symbols in the source JCL or in the START command for the task. For example, suppose you want to start CICS on SYSA and SYSB. To start CICS on each system, you can issue the command S CICS,JOBNAME=CICS&SYSNAME. This results in:
  - S CICS,JOBNAME=CICSSYSA on system SYSA
  - S CICS,JOBNAME=CICSSYSB on system SYSB

  Note: You can use system symbols in started task JCL (for both jobs and procedures) and in TSO logon procedures; however, you cannot specify system symbols in JCL for batch jobs. The reason for this is that system symbols are unique to each system. When you submit a job in a MAS, you have no idea which system (SYSA or SYSB) will do the JCL conversion for that job. So, if the conversion is done on SYSA, the system symbols are converted based on the value of the symbols on SYSA, but the job may end up running on SYSB.

There is no official mechanism for changing symbols without an IPL. However, there is an unsupported utility called SYMUPDTE available that permits symbols to be changed without an IPL. You can download the symupdte.exe file (program and documentation) from:

ftp://www.redbooks.ibm.com/redbooks/SG245451/

To be sure that the symbols are correctly spelled and set according to your needs, we recommend that you use the symbol syntax checker tool, a member of SYS1.SAMPLIB. Further information about how to install and use it, as well as its limitations, is available in the Appendix B, “Symbolic Parmlib Parser” of *z/OS MVS Initialization and Tunning Reference,* SA22-7592.

If you only need to verify a new Parmlib member’s use of the current system symbols, you can run the IEASYMCK sample program to see how the contents of the Parmlib member will appear after symbolic substitution occurs. IEASYMCK is located in SYS1.SAMPLIB. See the program prolog for details.

You can also enter the DISPLAY SYMBOLS operator command to display the static system symbols and associated substitution texts that are in effect for a system. See *MVS System Commands,* SA22-7627 for information about the use of this command.

Starting with DFSMS 1.4, you could use system symbols, for example &SYSR1, to represent volsers in the catalog entries of non-VSAM, non-SMS data sets. This is known as Indirect Volume Serial Support. With DFSMS/MVS 1.5 and later, it is also possible to use system symbols in extended alias definitions for data set names. These can then be related to different real data set names on different systems. This can be useful when migrating from one software release to the next. The advantage of this function is that your JCL does not have to be modified with respect to the data set names that you wish to use. What you must consider is the system on which you run the job. Selecting the system on which you run the job can be accomplished by the use of WLM Scheduling Environments. For more information, see Chapter 4, “WLM considerations” on page 51.

Some customers that use system symbols extensively have come up against the limit in the number of system symbols—prior to z/OS 1.4, you could have about 98 system symbols. In z/OS 1.4, the number of system symbols has been increased to at least 800 (the actual number depends on the size of the symbol names and the size of their values).

10.3 Sharing sysres

There are a number of reasons why you would want to consider sharing the target sysplex sysres with the incoming system. The first of these is that sharing your sysres between multiple systems gives you the benefit of having fewer images to manage and update. Along with the task of updating all those volumes, an even more labor-intensive task is trying to keep track of every sysres and what level it is at. So, if you can reduce the number of sysres', this can have a significant impact on the amount of time required to maintain all this information. Reducing the number of sysres also provides fewer points of failure.

The largest drawback of sharing it is that you have a single point of failure - if you lose the device, all systems that are IPLed off that device will come down. However, with modern DASD technology it is unlikely that you will lose a single volume—and if you lose a whole DASD subsystem, your system will more than likely come down anyway.

If you have very high availability requirements, an option is to have two physical sets of sysres volumes that are mirror images of each other. You can then IPL half the systems from one set of sysres' and IPL the other half from the other set. This way you avoid the single point of failure without having the overhead of having to maintain two full sets of sysres'.

10.3.1 Considerations for merging sysres'

If your target is a BronzePlex, then you have nothing to do as you are not sharing your sysres. If your target environment is a GoldPlex, we assume you will share the sysres and therefore get the benefits described. And if your target is a PlatinumPlex, you will again be sharing the sysres and continue to get the benefits. Table 10-1 shows what you need to consider, depending on what your target environment is.

Table 10-1 Considerations for merging sysres

Consideration	Note	Type	Done
Review performance considerations of sharing the sysres volume	1	G, P	
Ensure the sysres only contains sysres-eligible data sets	2	G, P	
Check that the incoming system and target sysplex have the same relative data set placement on their sysres volumes	3	G, P	
Check that the incoming system and target sysplex sysres' have the same product set and releases	4	G, P	
Plan for data set placement in the next ServerPac	5	G, P	
Both the incoming system and the systems in the target sysplex should have a similar change frequency	6	G, P	
Ensure there are available logical paths to the sysres' for the incoming system	7	G, P	
Using shared sysres to implement changes	8	G, P	

The "Type" specified in Table 10-1 relates to the sysplex target environment—**B** represents a BronzePlex, **G** represents a GoldPlex, and **P** represents a PlatinumPlex.

Notes indicated in Table 10-1 are described below:

1. If the sysres has a high access rate, performance is an important point to consider.

FICON® attachment is preferable as this provides support for higher I/O rates and more logical paths. The performance through one FICON attachment should provide approximately 4 to 5 times the throughput received with an ESCON connection.

Also, you should consider the use of parallel access volumes (PAV) and control units with large caches. For more information about PAV, refer to 4.3, “Considerations for merging WLMplexes” on page 54.

You should also aggressively use facilities such as LLA and VLF to minimize accesses to libraries on the sysres.

Contention, and therefore performance, is most likely to be an issue during a sysplex IPL, when many systems are IPLing, and therefore accessing the sysres, at the same time. However, sysplex IPLs should be a very rare event if you follow our recommendations for rolling IPLs.

2. The sysres should contain only data sets that meet the following criteria:
   - All the data sets must be read only.
   - The data sets should all be maintained *only* by SMP/E. Any data sets that are updated manually, for example SYS1.PARMLIB, should be placed on a different volume.

     Note: If you will be using SYS1.PARMLIB to contain your own customized members, make sure you remember to update the SMP/E PARMLIB DDDEF to point at SYS1.IBM.PARMLIB (which *should* remain on the sysres).

   - There should not be anything on the volume that is specific to one system or, indeed, one sysplex (so the same logical volume can be cloned across multiple operating environments). If you follow our naming conventions, all system-specific data sets would have the system name as part of the data set name, and be placed on a volume other than the sysres.
   - The sysres should only contain data sets that will be replaced by your next ServerPac. Otherwise, you are faced with the chore of moving those data sets manually when you migrate to the next operating system release.

   See the section entitled “Recommended data set placement” in *z/OS Planning for Installation,* GA22-7504. This information helps you decide which data sets should be placed on each of your sysres volumes.
3. Your objective should be to be able to move to the shared sysres and back again (if necessary) transparently. In order to do this (without having to change Master Catalog entries), you must ensure that all sysres data sets are placed on the same relative volumes, and that they are all cataloged indirectly. If the Master Catalog contains an entry saying that PRODUCT.V1R1M1.LOAD is on the second sysres volume, that entry will point at either the second sysres volume of your incoming system sysres or the second sysres volume of your target sysplex sysres, depending on which sysres you IPLed from. If this is not the case, you will get a “data set not found” message on some of the systems.

   You can check this by searching the output from an IDCAMS LISTCAT ALL for any occurrences of the sysres volume serial number. If you find any, either change them to be indirect, or move the data set off the sysres if it does not belong there. You can compare the contents of each sysres volume easily using ISPF 3.4 and SUPERC to compare the contents.

   Refer to 10.4, “Sharing Master Catalogs” on page 150 for more information about indirect volume serial support.
4. If you are going to share a sysres between a number of system images, you must ensure that the sysres you plan to use contains all the products that will be required on each

system. Presumably you will be changing the incoming system to use the sysres of the target sysplex. In this case, you must check that that sysres has the products and releases that are used on the incoming system.

5. You should also think ahead to the next ServerPac that you will install. If the same Master Catalog will be used to access both the old and new sysres', you must make sure that the new ServerPac places data sets on the same relative volumes. For this reason, you should make sure you allow enough free space on the sysres volumes for normal release-to-release growth of the product libraries.
6. All systems that are sharing the sysres should have the same maintenance and release policy. For example, if you have a system that gets updated with every release of the operating system, that would not be a suitable candidate for sharing a sysres with a system that gets updated less frequently.
7. The number of images that can share a sysres is limited predominately by the number of logical paths to the DASD device on which it resides.
8. We recommend that you have an alternate sysres on which you apply the maintenance. After applying maintenance, IPL one system from the alternate sysres and perform tests to ensure that everything is functioning correctly. This way, if there is a problem with the new sysres, only one system will be impacted, and you can easily fall back to the previous version of the sysres. Once you are satisfied that everything is functioning correctly, move the remaining systems over to that sysres in as short time as your schedules permit.

10.3.2 Methodology for moving to a shared sysres

Once you have completed all the checks in 10.3.1, "Considerations for merging sysres'" on page 147, the sysres for the incoming system should contain the same products, the same data sets (on the same relative sysres volume), and be at roughly the same release and service level as the sysres being used by the target sysplex systems.

You now have a choice as to when you bring the incoming system up using the target sysplex sysres. If you have followed our recommendations, the sysres should only contain read-only data sets, so Reserves should not be an issue. Therefore, if you wish to do so, it should be possible to IPL the incoming system from the target sysplex sysres *before* the incoming system actually joins the sysplex. The advantage of this approach is that you know that the incoming system is at the same release and maintenance level as all the other systems in the sysplex, and therefore you should not have to be concerned about toleration or compatibility maintenance when the incoming system joins the sysplex. You will also have the opportunity to make sure that the incoming system functions correctly with the target sysplex release and service levels *before* it joins the sysplex.

Note: If you decide to move to the new sysres before moving the incoming system into the sysplex, you must remember that you will be using a different Master Catalog to reference the sysres than all the other users of that sysres. Therefore, it is vital to be sure that all the sysres data sets cataloged in the incoming system's Master Catalog are cataloged on the correct relative sysres volume in advance of the move to the new sysres.

Another option is to bring the incoming system up using the target sysplex sysres at the time it joins the sysplex. This has the advantage of removing concerns about toleration and compatibility maintenance; however, it introduces yet another change into the cutover. Because you will already be making a significant number of changes at the time the incoming system joins the sysplex, it may be wiser not to also move to a new sysres at this time. As with any change or cutover, you will decrease the likelihood of problems if you can minimize the number of things being changed at one time.

Finally, you could bring the incoming system into the sysplex using its old sysres and then move to the target sysplex sysres at some later time—hopefully, not too long after the cutover. The disadvantage of this approach is that the incoming system may be at a different service or release level than the other systems in the sysplex, meaning that you may need to apply some toleration or compatibility service. Also, you may not have had the opportunity of testing that mix of release or service levels together in the sysplex, whereas if you move to the new sysres *before* the cutover, at least you know that the release and service levels being used by all the systems in the sysplex have been successfully tested together prior to the cutover.

10.4 Sharing Master Catalogs

Undoubtedly the largest benefit of a shared Master Catalog is the removal of all the work involved in trying to keep multiple Master Catalogs synchronized. Even if you follow the recommendations and minimize the number of data sets cataloged in the Master Catalog, it is still a considerable effort to keep the aliases and user catalogs synchronized across multiple Master Catalogs. In addition, because new products and new releases are provided at an increasing rate, there is continued pressure to upgrade systems more frequently than was the case in the past. The use of a shared Master Catalog helps to decrease the manual work involved with implementing those upgrades, especially for customers with large numbers of systems, each with their own Master Catalog.

The considerations for a shared Master Catalog with the possible target environments are as follows:

- In a BronzePlex, each system has its own Master Catalog. This means that each of them can be updated independently, increasing the possibility for conflicting changes and consequential errors.
- In a GoldPlex, the Master Catalog could be shared, but not all the user catalogs. Whether the Master Catalog is shared or not depends on how easy this would be to do. If both the Master Catalogs contain duplicate user data aliases (as opposed to aliases related to system software) that are pointing to different user catalogs (that is, you have duplicate data sets), you probably want to keep them separate. But, if your Master Catalogs contain only system-related entries, plus the user catalogs and user aliases, merging the catalogs should not be difficult, and the resulting ease-of-maintenance should justify the effort.
- In a PlatinumPlex, you have a single Master Catalog shared among all the systems of the sysplex. In this case, the amount of work involved in migrating or updating your Master Catalog when you install new software or a new ServerPac is reduced. This is the recommended scenario.

Before discussing the considerations, let us briefly review some concepts related to catalogs:

- A shared catalog is a basic catalog structure which is eligible to be used by more than one system. It was defined with SHAREOPTIONS(3 4), and resides on a volume that has been defined in HCD as shared. When a catalog with the correct SHAREOPTIONS is on a volume defined as SHARED, the catalog address space (CAS) will check to see if the in-storage buffers need to be refreshed before every catalog access. This ensures that each system is using an up-to-date copy of the catalog records at all times.
- Enhanced Catalog Sharing (ECS) is a sharing protocol that uses the CF to store the change information for a shared catalog. This information was previously stored in the VVDS, requiring at least one I/O for every shared catalog access. ECS provides catalog performance equivalent to a non-shared environment.

  For more information on ECS, see *z/OS DFSMS:Managing Catalogs,* SC26-7409 and the IBM Redbook *Enhanced Catalog Sharing and Management,* SG24-5594.

- Extended Alias Support is used to provide the ability to have only one alias for all systems, but have different libraries used, depending on which system the job runs. A parameter for the DEFINE ALIAS command, SYMBOLICRELATE, allows system symbols to be used in the specification of the base data set name.

  Suppose you have two systems, SYSA and SYSB, and the IEASYMxx Parmlib member definition shown in Example 10-1:

Example 10-1 Sample of IEASYMxx Parmlib member

```
SYSDEF          SYSCLONE(&SYSNAME(3:2))
                SYMDEF(&CLOCK='VM')              /* USE CLOCKVM   */
                SYMDEF(&COMMND='00')             /* USE COMMND00  */
                SYMDEF(&LNKLST='C0,C2')          /* LNKLST        */
                SYMDEF(&VATLST='00')             /* VATLST        */
                SYMDEF(&SMFPARM='00')            /* POINT TO SMFPRM00 */
                SYMDEF(&SSNPARM='00')            /* POINT TO IEFSSN00 */
                SYMDEF(&BPXPARM='FS')            /* SYSPLEX FILE SHARING */
SYSDEF          HWNAME(SCZP702)
                LPARNAME(A01)
                SYSNAME(SYSA)
                SYSPARM(00,01)
                SYMDEF(&SYSNAM='SYSA')
                SYMDEF(&SYSR2='&SYSR1(1:3).RS2') /* 2nd SYSRES logical ext */
                SYMDEF(&SYSR3='&SYSR1(1:3).RS3') /* 3rd SYSRES logical ext */
                SYMDEF(&SYSID1='1')
                SYMDEF(&OSZOSREL='ZOSR12')
                SYMDEF(&PRODVR='V1R3M0')
                SYMDEF(&CLOCK='00')              /* USE CLOCK00   */
SYSDEF          HWNAME(SCZP702)
                LPARNAME(A02)
                SYSNAME(SYSB)
                SYSPARM(00,01)
                SYMDEF(&SYSNAM='SYSB')
                SYMDEF(&SYSR2='&SYSR1(1:5).2') /* 2nd SYSRES logical ext */
                SYMDEF(&SYSR3='&SYSR1(1:5).2') /* 3rd SYSRES logical ext */
                SYMDEF(&SYSID1='2')
                SYMDEF(&OSZOSREL='ZOSR13')
                SYMDEF(&PRODVR='V1R4M0')
                SYMDEF(&CLOCK='00')              /* USE CLOCK00   *
```

  Note that for SYSA, the symbol &PRODVR. is defined as V1R3M0 and for SYSB, &PRODVR. is defined as V1R4M0.

  There is only one alias in the Master Catalog, created using the SYMBOLICRELATE parameter:

```
DEFINE ALIAS (NAME(SYS1.PRODUCT) SYMBOLICRELATE('SYS1.&PRODVR..PRODUCT'))
```

  This single alias definition has the ability to refer to different data sets, depending on which system the job runs. The result is as follows:

  - For SYSA: alias SYS1.PRODUCT refers to SYS1.V1R3M0.PRODUCT
  - For SYSB: alias SYS1.PRODUCT refers to SYS1.V1R4M0.PRODUCT

  In this case, the alias name is resolved at the time of use, rather than at the time of definition. When sharing systems are ready to upgrade to newer versions of the product (new data sets), they only need to change the definition of the appropriate system symbol or symbols to access the new data set by the same alias.

  For further information, refer to *z/OS DFSMS:Managing Catalogs,* SC26-7409.

- Indirect volume serial support allows the system to dynamically resolve volume and device type information for non-VSAM data sets that reside on either the system residence volume (sysres) or one or more logical extensions to the sysres volume. If all the sysres data sets do not fit on a single DASD volume, you can use additional volumes as logical extensions to sysres and refer to them indirectly. This allows you to change the volume serial number or device type of sysres or its logical extension volumes without having to recatalog the non-VSAM data sets on that volume.
  - For data sets that reside on the sysres volume, you can specify the indirect volume serial either:
    - A string of six asterisks (******) in place of the volume serial, or
    - The &SYSR1 static symbol in place of the volume serial

    The system dynamically resolves ****** or &SYSR1 to the volume serial of the volume from which the system was IPLed (the sysres volume).
  - For data sets that reside on the logical extensions to sysres, you must specify a static system symbol in place of the volume serial. This static symbol must be defined in the IEASYMxx Parmlib member.

  Using indirect catalog entries, together with the extended alias support, allows you to share a Master Catalog among multiple images that use different volumes with different names for the sysres volumes and their extensions. You can also do this using a single SYMDEF for all images in a shared Parmlib data set. Thus, once set up, no future updates should be needed to continue using this support.

  If your installation IPLs with different sysres volumes and you establish a naming convention for the sysres and its logical extension volumes, you can create a single IEASYMxx Parmlib member that can be used regardless of which sysres volume is used to IPL the system. To do this, use substrings of the sysres volume serial (&SYSR1) in defining the symbols for the extension volume serials (&SYSR2, &SYSR3, and so on). For example, assume you have the following sysres volumes and logical extensions listed in Table 10-2.

Table 10-2 Sysres volumes and logical extensions

System	IPLs from sysres	Has these extensions
SYS1	S01RES	S01RS2,S01RS3
SYS2	S02RES	S02RS2,S02RS3
SYS3	DEVRES	DEVRS2, DEVRS3

You can refer to them using a single IEASYMxx Parmlib member with the following statements:

Example 10-2 Sample definition for symbols &SYSR2 and &SYSR3

```
SYSDEF          SYSCLONE(&SYSNAME(3:2))
                SYMDEF(&CLOCK='VM')              /* USE CLOCKVM   */
                SYMDEF(&COMMND='00')             /* USE COMMND00  */
                SYMDEF(&LNKLST='C0,C2')          /* LNKLST       */
                SYMDEF(&VATLST='00')             /* VATLST       */
                SYMDEF(&SMFPARM='00')            /* POINT TO SMFPRM00 */
                SYMDEF(&SSNPARM='00')            /* POINT TO IEFSSN00 */
                SYMDEF(&BPXPARM='FS')            /* SYSPLEX FILE SHARING */
                SYMDEF(&SYSR2='&SYSR1(1:3).RS2') /* 2nd SYSRES logical ext */
                SYMDEF(&SYSR3='&SYSR1(1:3).RS3') /* 3rd SYSRES logical ext */
```

Note: The system automatically sets &SYSR1 to the volume serial of the IPL volume. You cannot define &SYSR1.

For details about how to use indirect volume serial support, see *z/OS MVS Initialization and Tunning Reference,* SA22-7592.

- Global Resource Serialization (GRS)

  The SYSIGGV2 reserve is used to serialize the entire catalog BCS component across all I/O, as well as to serialize access to specific catalog entries. The SYSZVVDS reserve is used to serialize access to associated VVDS records.

  The SYSZVVDS reserve along with the SYSIGGV2 reserve provide an essential mechanism to facilitate cross-system sharing of catalogs, and it is critical that these reserves are handled correctly in the GRS RNL. See item 10 on page 156 for more information about this.

10.4.1 Considerations for merging Master Catalogs

This section contains, in checklist format, a list of things that must be considered should you decide to merge the Master Catalogs. If your target is BronzePlex, then presumably you have nothing to do as you are not sharing your Master Catalog.

Table 10-3 Considerations for merging Master Catalogs

Consideration	Note	Type	Done
Check that the Master Catalogs are “clean”	**1**	**G, P**	
Check if any toleration service is required	**2**	**G, P**	
Confirm that the current Master Catalogs have similar characteristics	**3**	**G, P**	
If using ECS, ensure that the structure is large enough for the target number of catalogs using that facility	**4**	**G, P**	
Check if ECS can be enabled for additional catalogs as a result of the merge	**5**	**G, P**	
Confirm that the multi-level alias value is set the same in both systems	**6**	**G, P**	
Review RACF access to the catalogs	**7**	**G, P**	
Review use of cache features (ISC or CDSC - VLF) for catalogs	**8**	**G, P**	
Review GRS RNLs of target sysplex	**9**	**G, P**	
Review/update catalog backup and recovery procedure	**10**	**G, P**	

The “Type” specified in Table 10-3 relates to the sysplex target environment—**B** represents a BronzePlex, **G** represents a GoldPlex, and **P** represents a PlatinumPlex.

Notes indicated in Table 10-3 are described below:

1. Ideally a Master Catalog should contain as few entries as possible. It should contain:
   – All the data sets on the sysres volumes, plus other required system data sets (SMF, Parmlib, Proclib, page data sets, and so on). This should basically be just the data sets that were delivered on the ServerPac.

- User catalogs
- Aliases pointing to the user catalogs
- Possibly the distribution libraries that were delivered with the ServerPac

Having a “clean” Master Catalog makes the merge easier, simply because there are fewer entries to be merged. It also makes catalog recovery easier, should the catalog ever get damaged. And it provides better performance than a catalog with thousands of entries.

2. If the incoming system system is at the same DFSMS release and service level as the target sysplex, there are no considerations. However, if one of the systems is at a different release of DFSMS, you must ensure that any required toleration service has been applied. Information about the requirements for each release of DFSMS can be found in *z/OS DFSMS Migration,* GC26-7398.
3. You can issue the **MODIFY CATALOG,ALLOCATED** operator command to verify the characteristics of each of the Master Catalogs. See Example 10-3.

Example 10-3 MODIFY CATALOG,ALLOCATED operator command

```
F CATALOG,ALLOCATED
IEC351I CATALOG ADDRESS SPACE MODIFY COMMAND ACTIVE
IEC348I ALLOCATED CATALOGS 770
*CAS*************************************************************
* FLAGS -VOLSER-USER-CATALOG NAME                                *
* Y---R- #@$#M2 0001 MCAT.V#@$#M2                                *
* Y-I-R- #@$#I1 0001 UCAT.V#@$#I1                                *
* YSI-R- #@$#A1 0001 UCAT.V#@$#A1                                *
* Y-I-R- #@$#Q1 0001 UCAT.V#@$#Q1                                *
* Y-I-R- #@$#C1 0001 UCAT.V#@$#C1                                *
* Y-I-R- #@$#D1 0001 UCAT.V#@$#D1                                *
* Y-I-R- #@$#M1 0001 UCAT.V#@$#M1                                *
* Y-I-R- #@$#M1 0001 MCAT.V#@$#M1                                *
*****************************************************************
* Y/N-ALLOCATED TO CAS, S-SMS, V-VLF, I-ISC, C-CLOSED, D-DELETED, *
* R-SHARED, A-ATL, E-ECS SHARED, K-LOCKED                        *
*CAS*************************************************************
IEC352I CATALOG ADDRESS SPACE MODIFY COMMAND COMPLETED
```

4. You can share catalogs among systems which are running different levels of DFSMS, but you must be sure that the incoming system has the minimum DFSMS level required to support the features implemented on the target sysplex. For information on sharing catalogs between different levels of DFSMS, see *z/OS DFSMS Migration,* GC26-7398.
5. The Enhanced Catalog Sharing (ECS) structure name is SYSIGGCAS_ECS, and the structure size is related to the number of catalogs that will be enabled for ECS.

   You should use the CFSizer tool to determine how large the structure should be. The CFSizer is available at the following Web site:

   http://www.ibm.com/servers/eserver/zseries/cfsizer/
6. ECS cannot be used with catalogs that are shared outside the Parallel Sysplex. If the incoming system has catalogs that were previously shared with the systems in the target sysplex, then those catalogs were previously ineligible for use with ECS. If the only systems using those catalogs after the merge will all be in the new expanded sysplex, then you might consider enabling those catalogs for ECS after the merge.

Note: Although you can only have one ECS structure per sysplex (because the name is fixed), it is still possible to use ECS in a GoldPlex or even a BronzePlex, where half the DASD are offline to half the systems, and the remaining DASD are offline to the other half of the systems. It is acceptable to use ECS with a catalog that is unavailable to some of the systems in the sysplex, as long as you do not try to use the catalog from a system that does not have access to the volume containing the catalog and do not access the catalog from a system outside the sysplex.

7. A multilevel catalog alias is an alias of two or more high-level qualifiers.You can define aliases of up to four high-level qualifiers. Using multilevel aliases, you can have data sets with the same high-level qualifier cataloged in different catalogs, without using JOBCAT or STEPCAT DD statements. For example:

   a. Alias PROJECT1.TEST points to catalog SYS1.ICFCAT.PRO1TEST,

   b. Alias PROJECT1.PROD points to catalog SYS1.ICFCAT.PRO1PROD, and

   c. Alias PROJECT1 points to catalog SYS1.ICFCAT.PROJECT1.

   If the alias search level is 2, then data sets are cataloged as shown in Table 10-4.

Table 10-4 Multilevel Alias Facility

Data set	Catalog	Reason
PROJECT1.UTIL.CNTRL	SYS1.ICFCAT.PROJECT1	The second qualifier is neither TEST nor PROD.
PROJECT1.PROD.DATA	SYS1.ICFCAT.PRO1PROD	There are two qualifiers, and the two qualifiers form the alias PROJECT1.PROD.
PROJECT1.PROD	SYS1.ICFCAT.PROJECT1	There is only one qualifier in this data set name: PROJECT1.PROD is not a qualifier, so it is not used in the multilevel alias search.
PROJECT1.TEST.CNTRL	SYS1.ICFCAT.PRO1TEST	There are two qualifiers, and the two qualifiers form the alias PROJECT1.TEST.
PROJECT1.TEST.A.B	SYS1.ICFCAT.PRO1TEST	The first two qualifiers are used as the alias, since the search level is 2. The third qualifier is not used in the search.

   In this example, data being used for tests (TEST) is isolated from production programs and data (PROD) and other miscellaneous files. This isolation is desirable for data protection and availability. Backup and recovery of one catalog would not affect projects using the other catalogs.

   The alias search level is specified in the SYSCATxx member of SYS1.NUCLEUS or LOADxx member of SYS1.PARMLIB. It can also be changed without an IPL using the **MODIFY CATALOG,ALIASLEVEL** operator command. For more information, see *z/OS DFSMS:Managing Catalogs,* SC26-7409.

   For example, if the target sysplex has aliaslevel=2 and the incoming system has aliaslevel=3, you need to adapt the data set naming conventions of the incoming system to match the target sysplex's rules. In this case, you may need to merge some user catalogs, because you are not limited to only two levels in the aliases.

Note: While the aliaslevel *can* be set to a different value on each system, it *should* be set to the same value on every system that is sharing the affected catalogs. Using a different aliaslevel value on each system can lead to errors and inconsistent behavior, depending on where a job is run.

8. Make sure everyone has the correct RACF access to the new Master Catalog. One way to do this is to check the access list for the Master Catalog on the incoming system and ensure that those people also have the required access level to the new Master Catalog. We strongly recommend having a separate RACF profile for the Master Catalog, with a very restricted list of people that have UPDATE or higher access.

   To open a catalog as a data set, you must have ALTER authority. When defining an SMS-managed data set, the system only checks to make sure the user has authority to the data set name and SMS classes and groups. The system selects the appropriate catalog, without checking the user's authority to the catalog. You can define a data set if you have ALTER or OPERATIONS authority to the applicable data set profile.

   For more information, see “Chapter 5. Protecting Catalogs” in *z/OS DFSMS:Managing Catalogs,* SC26-7409.

9. Caching catalogs in storage is the simplest and most effective method of improving catalog performance. This reduces the I/O required to read records from catalogs on DASD.

   Two kinds of cache are available exclusively for catalogs:

   a. The in-storage cache (ISC) cache is contained within the Catalog Address Space (CAS). Its objective is to cache only those records that are read directly. It is the default and ideally would only be used for the Master Catalog. Since ISC is the default catalog cache, catalogs are cached in the ISC unless you specify that the catalog is to use CDSC, or unless you use the MODIFY CATALOG operator command to remove the catalog from the ISC.

   b. The catalog data space cache (CDSC) is separate from CAS and uses the MVS VLF component, which stores the cached records in a data space. You can add catalogs to the CDSC only by editing the COFVLFxx member to specify the catalogs, stopping VLF, then starting VLF. A sample COFVLFxx statement to add a catalog to CDSC is shown in Example 10-4.

Example 10-4 Defining a catalog to CDSC

```
CLASS NAME(IGGCAS) EMAJ(SYS1.SAMPCAT)
```

   Although you can use both types of catalog cache, you cannot cache a single catalog in both types of cache simultaneously. We recommend using CDSC for all user catalogs, and reserving the ISC for the Master Catalog.

   Example 10-3 on page 154 shows how to use the **`MODIFY CATALOG,ALLOCATED`** operator command to verify how a catalog is being cached.

   For more information, see Caching Catalogs section on *z/OS DFSMS:Managing Catalogs,* SC26-7409.

10. Verify that SYSZVVDS is defined in the GRS RNL EXCLUSION list. There should be a generic entry in the CONVERSION list for SYSIGGV2. In addition, if any catalogs are shared with systems outside the sysplex, you must include an entry in the EXCLUSION list with a QNAME of SYSIGGV2 and an RNAME of the catalog name. For example:

    ```
    RNLDEF RNL(EXCL) TYPE(SPECIFIC) QNAME(SYSIGGV2) RNAME(ucat.fred)
    ```

    For further information concerning catalog serialization, see the section entitled “VSAM and ICF Catalogs” in *z/OS MVS Planning: Global Resource Serialization,* SA22-7600.

11. To protect yourself from failures, we recommend that you have a procedure for recovering a Master Catalog from a backup, and also have a minimal Master Catalog that can be used to bring a system up that can be used to run the restore. The minimal Master Catalog should contain entries for all the data sets you need to IPL your system, logon to your TSO ID, and run the catalog recover job. You should create a SYSCATxx member in SYS1.NUCLEUS containing the name of the backup catalog, or maintain an alternate LOADxx member specifying the backup catalog name.

    To use the alternate SYSCATxx member, during the IPL, specify an initialization message suppression indicator (IMSI) on the LOAD parameter that tells the system to prompt for the Master Catalog response. You will then be prompted with a message requesting the two-character suffix of the SYSCAT member you wish the system to use.

    You should also review *z/OS DFSMS:Managing Catalogs,* SC26-7409 for a discussion about maintaining a backup of the Master Catalog.

10.4.2 Methodology for merging Master Catalogs

In this section we discuss the approach that can be taken for moving to a single shared Master Catalog. This process assumes that you will eventually use the target sysplex Master Catalog as the Master Catalog for all systems in the sysplex, and discard the Master Catalog from the incoming system.

The first step is to clean up the incoming system Master Catalog. Remove any entries that should not be there. As a general guideline, you should attempt to only have the following entries in your Master Catalog:

- Data sets on your sysres volumes. All these data sets should be indirectly cataloged.
- All SYS1. data sets.
- Page data sets.
- CDSs. Both the primary and alternate CDSs should be cataloged in the Master Catalog.
- User catalogs.
- Aliases.

There are two approaches for merging the Master Catalogs:

1. Compare the contents of both Master Catalogs (user catalog entries, aliases, etc.) to be sure that they are the same. In particular, check that duplicate alias entries are pointing to the same user catalogs. You would expect to find duplicate entries for the sysres data sets—this is not a concern. You probably will not find duplicate entries for the incoming system page, SMF, DAE, and similar data sets, so you must remember to add entries for these data sets to the target sysplex Master Catalog. You can use the MCNVTCAT program (available in the ServerPac SCPPLOAD library) and SUPERC to help in the task of reviewing and synchronizing the catalogs. If you need to merge user catalogs (perhaps because of different multilevel alias settings in the two systems), there is a merge process described in “4.2.2 Merging Catalogs” in *z/OS DFSMS:Managing Catalogs,* SC26-7409.
2. Repro the target sysplex Master Catalog to a brand-new, empty catalog. Then merge the incoming system Master Catalog in on top of that. This way, you do not change either of the running Master Catalogs. If you have any problem, at least you have something to fall back to. If you decide to use this approach, you must ensure that there are no updates to either of the Master Catalogs from the time you do the merge until you cut over to the new Master Catalog. Also, remember to create a new LOADxx or SYSCATxx member on all systems to point to the new Master Catalog.

Whichever method you choose, you also need to *keep* the catalogs in sync once you get them there. Fortunately the Master Catalog should not be updated frequently, and there should only be a small number of people with update access. Therefore, it should be possible to implement procedure changes to ensure that every time one Master Catalog is updated, the other one is also updated at the same time. You should also implement a regularly scheduled job to check the two catalogs to ensure they remain in sync, and identify any differences that may appear.

Having synchronized the Master Catalogs, the next thing to decide is when to change the incoming system to use the target sysplex Master Catalog. While it is possible to share a Master Catalog (or any catalog) between systems that are not in the same sysplex, this relies on using Reserve Release to serialize accesses to the catalog. Even though the Master Catalog typically has a very low rate of updates, we still feel it is best to completely avoid Reserves against the Master Catalog, if at all possible. Therefore, we strongly recommend waiting until *after* the cutover to switch the incoming system to the target sysplex Master Catalog.

10.5 Sharing Parmlib

The ideal scenario is the same for Parmlibs—it is more efficient if you can have just one Parmlib, making use of the system symbols. For more information about system symbols, refer to 10.2, “System symbols” on page 144.

The considerations for the various target environments are as follows:

- In a BronzePlex, you will only be sharing the sysplex CDSs, so there is no need to share the Parmlib. However, because the sysplex CDSs are pointed to from the COUPLExx member, you *might* wish to set up a data set in the Parmlib concatenation that contains *only* the COUPLExx member, and is shared by *all* systems in the sysplex.
- In a GoldPlex, you would probably share the Parmlib. If you use system symbols judiciously, it should be possible to have just one copy of each member, and have that member coded so that it can be used by every system. Even if some of the incoming system’s members are completely different, it is possible to have more than one version of every Parmlib member, and just use a different suffix to pick up that member on the incoming system.
- In a PlatinumPlex, you would share everything—Parmlib, Proclib, and TSO LOGON and CLIST libraries. In some cases, this may require additional work up front, however the reduced maintenance in the long term should more than pay back the time invested in this task.

  To help you move to a single Parmlib, the use of system symbols, while not mandatory, is certainly *highly* recommended. Table 10-5 lists all Parmlib members that support system symbols. It can help in the task of determining what are the system symbols used by the target sysplex’s Parmlib against those used in the incoming system’s Parmlib.

Table 10-5 Parmlib members that support system symbols

Parmlib member	Function
ADYSETxx	Parameters that control dump analysis and elimination (DAE) processing.
ALLOCxx	Parameters that control installation defaults for allocation values.
APPCPMxx	Parameters that define or modify the APPC/MVS configuration.
ASCHPMxx	Parameters that define scheduling information for the ASCH transaction scheduler.

Parmlib member	Function
BPXPRMxx	Parameters that control the z/OS UNIX System Services environment and the hierarchical file system (HFS). The system uses these values when initializing the z/OS UNIX System Services kernel.
CLOCKxx	Parameters that control operator prompting to set the TOD clock, specifying the difference between the local time and GMT, and ETR usage.
CNGRPxx	Parameters that define console groups as candidates for switch selection if a console fails.
COFDLFxx	Allows a program to store DLF objects that can be shared by many jobs in virtual storage managed by Hiperbatch™.
COFVLFxx	Allows an authorized program to store named objects in virtual storage managed by VLF.
COMMNDxx	Commands to be issued by the control program immediately after initialization. JES commands may not be included. For more information, see 10.5.2, “COMMNDxx Parmlib member” on page 162.
CONFIGxx	Allows the installation to define a standard configuration that is compared with the current configuration and to reconfigure processors, storage, and channel paths.
CONSOLxx	Parameters to define an installation’s console configuration, initialization values for communications tasks, the default routing codes for all WTO/WTOR messages that have none assigned, and the characteristics of the hardcopy message set. CONSOLxx also contains parameters that define the hardcopy medium and designate the alternate console group for hardcopy recovery.
COUPLExx	Describes the sysplex environment and CDSs.
CSVLLAxx	Allows an installation to list the entry point name or LNKLST libraries that can be refreshed by the MODIFY LLA, UPDATE=xx command.
CSVRTLxx	Defines the run-time library services (RTLS) configuration. CSVRTLxx can be used to specify names of libraries to be managed, as well as storage limits for caching modules from the libraries.
CTncccxx	Specifies component trace options.
DIAGxx	Contains diagnostic commands that control the common storage tracking and GETMAIN/FREEMAIN/STORAGE (GFS) trace functions.
EXITxx	Allows an installation to specify installation exits for allocation decisions.
EXSPATxx	Allows an installation to specify actions taken to recover from excessive spin conditions without operator involvement.
GRSCNFxx	Configuration parameters for systems that are to share resources in a global resource serialization complex.
GRSRNLxx	Resource name lists (RNLs) that the system uses when a global resource serialization complex is active.
IEAAPP00	Names of authorized installation-written I/O appendage routines.
IEACMD00	IBM-supplied commands that are processed during system initialization.
IEADMCxx	Parameters to control DUMP processing.
IEAFIXxx	Names of modules to be fixed in central storage for the duration of the IPL.

Parmlib member	Function
IEAICSxx	Parameters of an installation control specification that associate units of work (transactions) with performance groups. Performance groups are used by the system resources manager to control and report on transactions. This member is only used in WLM compatibility mode.
IEAIPSxx	Parameters of an installation performance specification that control the actions of the system resource manager. This member is only used in WLM compatibility mode.
IEALPAxx	Names of reenterable modules that are to be loaded as a temporary extension to the PLPA.
IEAOPTxx	Parameters that control resource and workload management algorithms in the system resource manager.
IEAPAKxx	"Pack List" names of groups of modules in the LPALST concatenation that the system will load between page boundaries to minimize page faults.
IEASLPxx	Contains SLIP commands.
IEASVCxx	Allows the installation to define its own SVCs.
IEASYMxx	Specifies, for one or more systems in a multisystem environment, the static system symbols and suffixes of IEASYSxx members that the system is to use. One or more IEASYMxx members are selected using the IEASYM parameter in the LOADxx Parmlib member.
IEASYSxx	System parameters. Multiple system parameter lists are valid. The list is chosen by the operator SYSP parameter or through the SYSPARM statement of the LOADxx Parmlib member (see Chapter 58, "LOADxx (System Configuration Data Sets)" *z/OS MVS Initialization and Tunning Reference,* SA22-7592 for more information).
IECIOSxx	Parameters that control missing interrupt handler (MIH) intervals and update hot I/O detection table (HIDT) values.
IEFSSNxx	Parameters that identify what subsystems are to be initialized.
IFAPRDxx	Parameters that define the product enablement policy.
IGDSMSxx	Initialize the Storage Management Subsystem (SMS) and specify the names of the active control data set (ACDS) and the communications data set (COMMDS) and other SMS parameters.
IKJTSOxx	For TSO/E, specifies authorized commands and programs that are authorized when called through the TSO service facility, commands that may not be issued in the background, and defaults for SEND and LISTBC processing.
IVTPRM00	Sets Communications Storage Manager (CSM) parameters.
LNKLSTxx	List of data sets to be concatenated to form the LNKLST concatenation. You can also use PROGxx to define the concatenation.
LOADxx	Specifies data sets MVS uses to configure your system. Supports &SYSR1 as a volser. No other system symbol support.
LPALSTxx	List of data sets to be concatenated to SYS1.LPALIB from which the system builds the pageable LPA (PLPA).
MMSLSTxx	Specifies information that the MVS message service (MMS) uses to control the languages that are available in your installation.

Parmlib member	Function
MPFLSTxx	Parameters that the Message Processing Facility uses to control message processing and display.
MSTJCLxx	Contains the master scheduler job control language (JCL) that controls system initialization and processing. For more information, see 10.5.1, “MSTJCLxx Parmlib member” on page 161.
PFKTABxx	Parameters contain the definitions for program function key tables (PFK tables).
PROGxx	Contains statements that define the format and contents of the APF-authorized program library list, control the use of installation exits and exit routines, define the LNKLST concatenation, and specify alternate data sets for SYS1.LINKLIB, SYS1.MIGLIB, and SYS1.CSSLIB to appear at the beginning of the LNKLST concatenation and SYS1.LPALIB to appear at the beginning of the LPALST concatenation.
SCHEDxx	Provides centralized control over the size of the master trace table, the completion codes to be eligible for automatic restart, and programs to be included in the PPT.
SMFPRMxx	Parameters that define SMF options.
TSOKEYxx	Parameters that are used by TSO/VTAM time sharing.
VATLSTxx	Volume attribute list that defines the “mount” and “use” attributes of direct access volumes.
XCFPOLxx	Specifies the actions to be taken if a system fails to update its system status in the sysplex CDS. This member is ignored if SFM is active.

10.5.1 MSTJCLxx Parmlib member

The MSTJCLxx Parmlib member contains the master scheduler job control language (JCL) that controls system initialization and processing. You can place the master scheduler JCL in MSTJCLxx as an alternative to keeping it in the MSTJCLxx module in SYS1.LINKLIB.

If the system finds a MSTJCLxx Parmlib member, it uses the JCL in that member to start the master scheduler address space. If the system does not find a MSTJCLxx Parmlib member, it uses the JCL in the MSTJCLxx module in SYS1.LINKLIB.

You can specify symbols in MSTJCLxx. Example 10-5 shows how you could have a common SYS1.PROCLIB, CPAC.PROCLIB, and SYS1.IBM.PROCLIB on all systems, and have each system also have a Proclib data set that is unique to that system.

Example 10-5 MSTJCL00 Parmlib member

```
//MSTJCL00 JOB MSGLEVEL=(1,1),TIME=1440
//         EXEC PGM=IEEMB860,DPRTY=(15,15)
//STCINRDR DD SYSOUT=(A,INTRDR)
//TSOINRDR DD SYSOUT=(A,INTRDR)
//IEFPDSI  DD DSN=SYS1.PROCLIB,DISP=SHR
//         DD DSN=CPAC.PROCLIB,DISP=SHR
//         DD DSN=SYS1.IBM.PROCLIB,DISP=SHR
//         DD DSN=SYS1.&SYSNAME..PROCLIB,DISP=SHR
//SYSUADS  DD DSN=SYS1.UADS,DISP=SHR
//SYSLBC   DD DSN=SYS1.BRODCAST,DISP=SHR
```

10.5.2 COMMNDxx Parmlib member

COMMNDxx is an optional, installation-created list of automatic commands the system internally issues as part of system initialization. COMMNDxx is useful for automatic entry of commands that are always issued at system initialization.

System commands in COMMNDxx can contain system symbols. System symbols can represent any type of variable text in system commands, with the exception of command prefixes and names.

Be aware that the system does not process system symbols in COMMNDxx during Parmlib processing. Instead, the system processes the system symbols in the same way that it processes system symbols in commands that are entered on a console. Example 10-6 contains a sample member.

Example 10-6 COMMNDxx Parmlib member

```
COM='SET MPF=00'
COM='S JES2,PARM=NOREQ'
COM='START NET,,,(LIST=&SYSCLONE),SUB=MSTR'
COM='START VLF,SUB=MSTR,NN=00'
COM='S NETVIEW,SUB=MSTR,CAT=&CATALOG.,SYS=&SYSNAME'
COM='S APPC,SUB=MSTR,APPC=02'
COM='SET DAE=&DAE'
```

10.5.3 Merging Parmlibs

You can use the SUPERC utility to help you analyze the members of both Parmlib libraries (from the incoming system and from the target sysplex), searching for members with duplicate names but different contents, and resolving the conflicts.

As an alternative, you can use Parmlib concatenation. You can define up to 10 partitioned data sets defined through Parmlib statements in the LOADxx member of either SYSn.IPLPARM or SYS1.PARMLIB. SYS1.PARMLIB makes the 11th (or last) data set in the concatenation, and is the default Parmlib concatenation if no PARMLIB statements exist in LOADxx. If you need to dynamically change the Parmlib concatenation, you can issue the SETLOAD command. For more information about the SETLOAD command, refer to *MVS System Commands,* SA22-7627.

Because synchronizing the incoming system's and target sysplexes' Parmlibs can be a time-consuming exercise, we recommend starting the merge as early as possible. You should aim to have the members of the incoming system's Parmlib identical to the corresponding members of the target sysplex, and to have all the required definitions for the incoming system in the target sysplex Parmlib before the incoming system is IPLed into the sysplex. Thus, at the time of the cutover, switching to the target sysplex Parmlib should be simply a matter of changing the IPL Load parameter.

We do *not* recommend switching to the target sysplex Parmlib in advance of the cutover because Reserves would need to be used to serialize access to Parmlib, and it is unwise to have any unnecessary Reserves issued against the Parmlib volume.

10.6 Proclibs and TSO-related files

It can be more efficient if you can have one Proclib, TSO LOGON, CLIST libraries and similar data sets. The use of system symbols can help to achieve this, although moving to a single consolidated set of Proclibs can be more difficult, especially if the incoming system is totally unrelated to the existing systems.

The first thing you must do is identify the members of Proclib that are being used by the incoming system. If you look at the members of Proclib, they usually belong to four categories:

- System-started tasks, like VLF and TCP
- Installation-specific tasks like CICS or IMS
- Procedures that are used by batch jobs
- Groups of JCL statements that get imbedded in batch jobs or started tasks by INCLUDE statements

If the existing systems in the target sysplex already share Proclib, the JCL for the system tasks should already be set up in a manner that they can be tailored for each system (probably through the use of system symbols). To extend those members to support the incoming system, compare the JCL in those members with the corresponding members in the incoming system. Hopefully all that should be required to support the incoming system is to specify the appropriate values for the system symbols used in those members.

For the members that contain the JCL for installation-specific tasks like CICS or IMS, you first need to verify that there is no duplication between those members and existing members in the target sysplex. If there are, you will probably have to change the started task names—if you do this, remember to also change the security definitions and automation routines that are keyed off the started task name. If there is no duplication, you simply need to ensure that the started task JCL will be in a library that is accessible to the incoming system after that system moves into the sysplex.

For the procedures and INCLUDE members, you have to check for duplication between the libraries on the incoming system and those in the target sysplex—hopefully there will be very little (if you have good naming conventions). If there are duplicates, you will need to address each of these on a case-by-case basis. If there are no duplicates, once again you simply need to ensure that the members will be in a library that is accessible to the incoming system after it joins the sysplex.

If you can keep SYS1.PROCLIB with only the members that IBM delivers, it makes system upgrades easier. Therefore try to place all your own procs in one or more user Proclibs and access them using one of the following options:

- You can use a JCLLIB statement in each job to point to a specific Proclib, or to a sequence of Proclibs.

  Assume that SYS1.PROCLIB is the only system default procedure library. If you do not specify the JCLLIB statement, then the system searches only SYS1.PROCLIB. If you specify JCLLIB, the system searches the libraries on the JCLLIB statement before SYS1.PROCLIB. Example 10-7 contains an example.

Example 10-7 Proclib concatenation

```
//MYJOB JOB     ...
//MYLIBS JCLLIB  ORDER=(SYSPLEX.PROCS.JCL,PLEX1.PROCS.JCL,
//            PLEX2.PROCS.JCL,PLEX3.PROCS.JCL)
//S2       EXEC    PROC=MYPROC
                .
```

```
                .
The system searches the libraries for procedure MYPROC in the following order:
1. SYSPLEX.PROCS.JCL
2. PLEX1.PROCS.JCL
3. PLEX2.PROCS.JCL
4. PLEX3.PROCS.JCL
5. SYS1.PROCLIB
```

- In the JES2 started task JCL, you can have multiple PROCxx statements, each potentially pointing to a different set of Proclibs. In the JES2PARM, you can say which PROCxx statement you want each job class to use. For example:

Example 10-8 JES2 started task JCL

```
//JES2    PROC
//IEFPROC EXEC PGM=HASJES20,TIME=1440,DPRTY=(15,14)
//HASPLIST  DD DDNAME=IEFRDER
//HASPPARM  DD DSN=SYS1.PARMLIB(J2USECF),DISP=SHR
//PROC00    DD DSN=SYS1.PROCLIB,DISP=SHR
//          DD DSN=CPAC.PROCLIB,DISP=SHR
//PROC01    DD DSN=SYS1.&SYSNAME..PROCLIB,DISP=SHR
//          DD DSN=SYS1.PROCLIB,DISP=SHR
```

- You can use system symbols in the MSTJCL member, so you can have a mix of shared and system-specific Proclibs. The IEFPDSI DD statement defines the data set that contains procedure source JCL for started tasks. Normally this data set is SYS1.PROCLIB, but it can be any library in the concatenation. For useful work to be performed, the data set must at least contain the procedure for the primary JES. For an example, refer to Example 10-5 on page 161.
- Another option is to use INCLUDE JCL statements. Example 10-9 contains an example.

Example 10-9 INCLUDE JCL

```
The following INCLUDE group is defined in member SYSOUT2 of private library
CAMPBELL.SYSOUT.JCL.
    //*  THIS INCLUDE GROUP IS CATALOGED AS...
    //*  CAMPBELL.SYSOUT.JCL(SYSOUT2)
    //SYSOUT2  DD      SYSOUT=A
    //OUT1     OUTPUT  DEST=POK,COPIES=3
    //OUT2     OUTPUT  DEST=KINGSTON,COPIES=30
    //OUT3     OUTPUT  DEST=MCL,COPIES=10
    //*  END OF INCLUDE GROUP...
    //*  CAMPBELL.SYSOUT.JCL(SYSOUT2)

The system executes the following program:
    //TESTJOB  JOB  ...
    //LIBSRCH  JCLLIB  ORDER=CAMPBELL.SYSOUT.JCL
    //STEP1    EXEC    PGM=OUTRTN
    //OUTPUT1  INCLUDE MEMBER=SYSOUT2
    //STEP2    EXEC    PGM=IEFBR14

The JCLLIB statement specifies that the system is to search private library
CAMPBELL.SYSOUT.JCL for the INCLUDE group SYSOUT2 before it searches any system libraries.

After the system processes the INCLUDE statement, the JCL stream appears as:
    //TESTJOB  JOB  ...
    //LIBSRCH  JCLLIB  ORDER=CAMPBELL.SYSOUT.JCL
    //STEP1    EXEC    PGM=OUTRTN
    //*  THIS INCLUDE GROUP IS CATALOGED AS...
    //*  CAMPBELL.SYSOUT.JCL(SYSOUT2)
```

```
//SYSOUT2  DD      SYSOUT=A
//OUT1     OUTPUT  DEST=POK,COPIES=3
//OUT2     OUTPUT  DEST=KINGSTON,COPIES=30
//OUT3     OUTPUT  DEST=MCL,COPIES=10
//*  END OF INCLUDE GROUP...
//*  CAMPBELL.SYSOUT.JCL(SYSOUT2)
//STEP2    EXEC    PGM=IEFBR14
```

Note: Both JCLLIB and INCLUDE are somewhat restrictive because you can't use system symbols in JCL.

The considerations for the various target environments are as follows:

- In a BronzePlex, you will only be sharing the sysplex CDSs, so there is no need to share the Proclib or TSO-related files.
- In a GoldPlex, you would probably share the Proclib.

  Sharing Proclibs and CLIST libraries is a little more complex. There are many system tasks that have their JCL in SYS1.PROCLIB, and *must* have a certain name. If the incoming system must have different JCL for some of those members, you may wish to retain separate Proclibs. Similarly, if you have similarly named TSO CLISTs on the different systems that do completely different things, you will probably keep them separate.
- In a PlatinumPlex, as mentioned before, you would share everything—Parmlib, Proclib, and TSO LOGON and CLIST libraries.

10.6.1 Merging Proclibs and TSO-related files

If you decide to go ahead with the merge, you can use the SUPERC utility to identify duplicate member names and compare member contents.

Because synchronizing the incoming system's and target sysplexes' Proclibs and TSO libraries can be a time-consuming exercise, we recommend starting the merge as early as possible. You should aim to make the members of the incoming system's Proclib that are being used identical to the corresponding members of the target sysplex, and to have all the corresponding members in the target sysplex set up to work with the incoming system after it is IPLed into the sysplex. Thus, at the time of the cutover, switching to the target sysplex Proclib should be simply a matter of coming up using those libraries.

We do *not* recommend actually switching to the target sysplex libraries in advance of the cutover because, as mentioned, Reserves would need to be used to serialize access to Proclib, and it is unwise to have any unnecessary Reserves issued against the Proclib volume.

10.7 Tools and documentation

In this section we provide lists of the tools and documentation mentioned in this chapter that may help you complete this task.

10.7.1 Tools

Table 10-6 on page 166 contains a list of tools that may help, the function of each tool, and where you can get further information about the tool.

Table 10-6 Tools

Tool	Function	Reference
SYMUPDTE utility	Permits symbols to be changed without an IPL. **Note:** This is an unsupported utility.	ftp://www.redbooks.ibm.com/redbooks/SG245451/
Symbol syntax checker tool	Verify that symbols are correctly spelled and set according to your needs.	SYS1.SAMPLIB member. For information about how to use it, refer to Appendix B. Symbolic Parmlib Parser of *z/OS MVS Initialization and Tunning Reference,* SA22-7592.
IEASYMCK	Verify how the contents of the Parmlib member will appear after symbolic substitution.	SYS1.SAMPLIB member.
Coupling Facility Structure Sizer Tool (CFSIZER) CFSIZER	Use to size the structures and generate the JCL.	http://www.ibm.com/servers/eserver/zseries/pso/
MCNVTCAT program and SUPERC	Compare the contents of the Master Catalog	MCNVTCAT is available in the ServerPac SCPPLOAD library.

10.7.2 Documentation

The following documentation may be helpful as you merge the entities discussed in this chapter:

- *MVS System Commands,* SA22-7627
- *z/OS DFSMS:Managing Catalogs,* SC26-7409
- *z/OS DFSMS Migration,* GC26-7398
- *z/OS MVS Initialization and Tunning Reference,* SA22-7592
- *z/OS MVS Planning: Global Resource Serialization,* SA22-7600
- *z/OS Planning for Installation,* GA22-7504

11

VTAM considerations

This chapter discusses the following aspects of moving a system that is using VTAM into a sysplex where one or more of the systems are also using VTAM:

- Is it necessary to merge VTAM environments, or is it possible to have more than one VTAM environment in a single sysplex?
- A checklist of things to consider should you decide to merge the VTAMs.
- A list of documentation that can assist with this exercise.

11.1 Overview of scope

While the overall intent of this book has been to look at three possible sysplex merge options (BronzePlex, GoldPlex or PlatinumPlex), from a VTAM perspective the options are somewhat simpler—either you want to connect VTAM on the incoming system to the VTAMs on the target sysplex systems, or you want to keep it separate.

It is expected that readers understand SNA Subarea and APPN network characteristics and configurations in some depth, are knowledgeable of their own environments, and also understand their own network merge requirements.

While touching briefly on the benefits of VTAM Generic Resources and MultiNode Persistent Sessions (MNPS), this chapter is not intended to help you understand these facilities.

Before discussing the considerations for the merge, it is appropriate here to summarize VTAM's role in general, and specifically how it uses the facilities available in a sysplex environment.

11.1.1 VTAM's job

A SNA network facilitates the establishment of sessions between users and applications. Terminals and applications in an SNA network are represented by logical unit (LUs), and each has a unique name in the form of network-id.lu-name.

It is the job of VTAM to provide network services to applications that communicate across an SNA network. The network services that VTAM provides are:

- An application programming interface (API) to enable data transfer, without needing to know anything about the physical network
- The ability to locate a session partner (LU) in the network by its network name, independent of where that partner resides
- Selection of the optimum route between session partners
- Maintenance of the session across the network once the session has been established (that is, provide error recovery, maintain throughput, and so on)

SNA has two different architectures: subarea networking and Advanced Peer-to-Peer Networking® (APPN). To the user and application program they appear the same, however, in VTAM they are quite different.

11.1.2 Subarea networking

A subarea network is hierarchical and network services are centralized. Every LU in a subarea network is defined to, and owned by, a VTAM. Each LU has to request all of its network services from the VTAM to which it is primarily connected.

The network topology is distributed, but static. All paths in a subarea network must be manually defined, and route selection is performed by choosing from predefined lists of routes.

Access to remote LUs need not be predefined on the local VTAM, as this can be determined by searching the network.

11.1.3 APPN networking

An APPN network is organized on a peer-to-peer basis. There is no hierarchy as there is in a subarea networking environment, apart from the ability to be able to distinguish network nodes (NNs), which provide the network services, and end nodes (ENs), which provide only the API and services for the local resources.

Because there is essentially no hierarchy, there is not the same range of node types defined. Effectively all APPN nodes are Type 2.1, and indeed this is how they appear to a subarea network.

The NNs provide the distributed resource directory in an APPN network and locating session partners is dynamic, without the need to predefine the owner of a resource. In a large network there is usually one or more central directory servers (CDSs), in which the distributed directory tends to be concentrated. The use of CDSs reduces search traffic because the location of target resources is more likely to be known to a CDS.

All VTAM systems in a sysplex must use APPN protocols if they wish to use XCF to communicate with each other or if they wish to use VTAM GR or MNPS. However, there is no such requirement outside of the sysplex where the VTAM sysplex functions are still available even when subarea protocols are still used. For example, refer to the configuration in Figure 11-1.

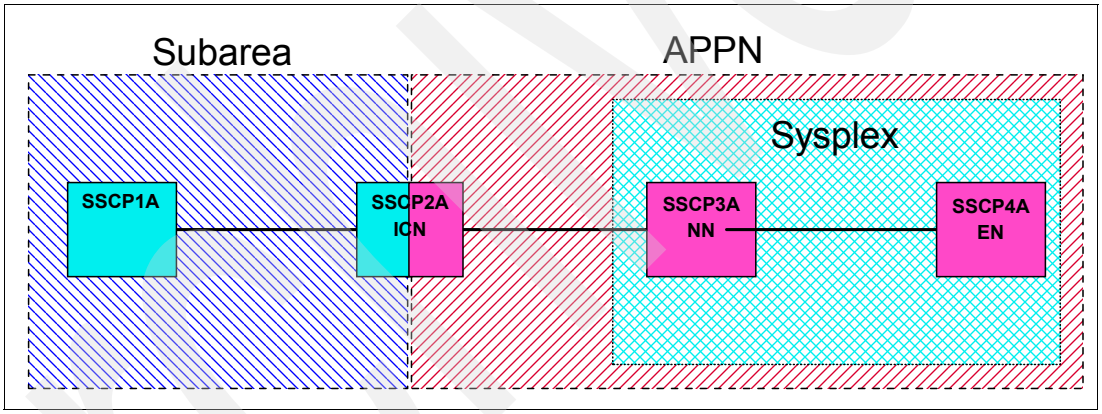


Figure 11-1 Subarea and APPN considerations

In this configuration, Nodes 1A and 2A are subarea-capable, and Nodes 2A, 3A and 4A are all APPN-capable. Node 2A acts as the Interchange Node (ICN) between the subarea and APPN networks. Within the APPN network, only Nodes 3A and 4A (and other ENs like 4A) are in the sysplex. Node 3A is the NN that is acting as the Network Node Server (NNS) for all of the ENs in the sysplex. Also, assume that Nodes 3A and 4A (and the other ENs) are all participating in the same “GRplex” and/or “MNPSplex”.

LUs which are owned by (and attached to) Node 1A (in the subarea network) can still establish sessions to GR and/or MNPS APPLs in the sysplex. When one of these LUs initiates a session to a GR-name or MNPS APPL, the session request is routed through subarea to 2A, then into the APPN network using the ICN function of 2A. During the APPN session setup process, the GR and/or MNPS functions are available to those LUs.

For GR, this means that when Node 3A receives the request, it will use the sysplex/XCF GR information to choose a GR-instance on one of the ENs, and the session will be established with that instance. If that instance should fail, and the LU attempts a new session to the GR-name, Node 3A will resolve this new session request to another still-working instance of that GR and a new session will be established.

For MNPS, this means that the session will be established to the MNPS APPL that was requested. If HPR was used for the APPN part of the session path and the host (EN) that the MNPS APPL is on fails, then the MNPS APPL can be restarted on one of the other ENs and the existing session can be PATH SWITCHED to the new owning host.

To summarize, the session path does not have to be *entirely* APPN in order to be able to establish sessions with GR or MNPS APPLs—only the sysplex portion of the session path must be APPN.

11.1.4 How VTAM uses XCF

When running in a sysplex, it is possible for VTAM to automatically connect with other VTAMs in the sysplex via XCF. This function is designed so that no VTAM definitions are required, making it easy to implement. Assuming that all the prerequisites are in place[1], VTAM will connect to XCF automatically during VTAM startup *unless you specifically indicate that this is not to happen*. Therefore—again presuming the prerequisites are met on every system—by default, every VTAM in the sysplex will connect to all the other VTAMs in the sysplex. The mechanism for this communication is an XCF group called ISTXCF.

VTAM dynamically creates TRLE and PU definitions for the other VTAMs that are also connected to the ISTXCF group. These connections are assigned unique names using the &SYSCLONE system variable values for the systems in the sysplex. The &SYSCLONE variable is typically defined in the IEASYMxx member in SYS1.PARMLIB.

If you do not wish this full connectivity to take place, it is possible to override this processing. You can prevent VTAM from joining the XCF group at startup by defining XCFINIT=NO in the startup options (ATCSTRxx). If, at a later stage, you wish to allow VTAM to join the XCF group you can issue a V NET,ACT,ID=ISTLSXCF. VTAM will not check for a definition of this name, but will instead automatically connect, as if XCFINIT=YES (the default) had been defined in the startup options during VTAM initialization.

There are a number of ways to determine if XCF is being used by VTAM. For example, you can display the ISTLSXCF node in VTAM, as shown in Example 11-1. Any response other than `ID PARAMETER VALUE netid.ISTLSXCF NOT VALID` means that VTAM is using XCF to communicate with other VTAMs in the sysplex.

Example 11-1 Displaying status of VTAM XCF interface

```
D NET,ID=ISTLSXCF,E
IST097I DISPLAY ACCEPTED
IST075I NAME = ISTLSXCF, TYPE = LCL SNA MAJ NODE 421
IST486I STATUS= ACTIV, DESIRED STATE= ACTIV
IST084I NETWORK RESOURCES:
IST1316I PU NAME = ISTP$1$2  STATUS = ACTIV---X-  TRLE = ISTT$1$2
IST1316I PU NAME = ISTP$1$3  STATUS = ACTIV---X-  TRLE = ISTT$1$3
IST1500I STATE TRACE = OFF
IST314I END
```

Another option is to display who is connected to the ISTXCF XCF group, as shown in Example 11-2 on page 171.

[1] All VTAM systems in the sysplex that wish to use VTAM's sysplex features must use the APPN protocols to communicate with each other; therefore, APPN must be implemented within the sysplex in this case.

Example 11-2 Displaying who is connected to the VTAM XCF group

```
D XCF,GROUP,ISTXCF
IXC332I  02.22.09  DISPLAY XCF 423
 GROUP ISTXCF:    #@$1M$$$USIBMSC   #@$2M$$$USIBMSC   #@$3M$$$USIBMSC
```

This shows the names of the three VTAMs in the sysplex and their network names. As you can see, #@$1M, the VTAM that is running on this image, is a member of the XCF group and therefore VTAM is currently using XCF services.

Note: If you wish the TCP/IP stack on a system to be able to exchange packets with other TCP/IP stacks on other systems in the sysplex via XCF links, VTAM *must* be connected to this group.

If the “incoming system” is actually a set of systems that currently use dynamic XCF links to communicate with each other, and your target environment is a BronzePlex (meaning that you don’t want *all* the VTAMs in the sysplex to communicate with each other), then you must change the incoming systems to use some method other than dynamic XCF links (such as statically-defined links) to communicate with each other.

11.1.5 VTAM Generic Resources

The VTAM Generic Resources (GR) feature provides the capability to improve availability and do load balancing across systems in a sysplex. This is achieved through the assignment of a generic name to a group of active applications providing the same function (for example, CICS).

Users log on to the resource using the generic name, and the VTAM owning the user’s session dynamically determines which instance of the application is the most appropriate one to route the logon request to. In this way, if one of the application instances becomes unavailable, the outage is shielded from the user, whose logon request is simply routed to a different instance of the application.

VTAM GR uses a structure in the CF to contain information about which application instances are connected to each generic resource name, and which LUs are in session with each instance. The structure name defaults to ISTGENERIC; however, it is possible to override this name on the STRGR start option.

If you do not wish a particular system to be able to use VTAM GR, and VTAM GR is in use on other systems (that is, the ISTGENERIC structure is defined), you must explicitly code STRGR=NONE in the VTAM start options.

Restriction: While VTAM provides the ability to specify a structure name other than ISTGENERIC, it is not recommended to use a different structure name on some systems in the sysplex in an attempt to create subplexes.

This capability was provided to give customers the ability to have a test VTAM GR group active within the same sysplex as a production VTAM GR group. While this could be viewed as providing a subplex capability (some systems could use one VTAM GR group and others could use another), this is not the intended purpose of this capability.

If the “incoming system” is actually a set of systems that currently use VTAM GR, and the existing systems in the target sysplex are also using VTAM GR, and your target environment is a BronzePlex (meaning that you don’t want *all* the VTAMs in the sysplex to communicate with each other), then you have a problem. While it is technically possible to have two different “VTAM GRplexes” in the sysplex, this is not recommended for production—meaning that one of the VTAMplexes is going to have to discontinue its use of VTAM GR.

11.1.6 MultiNode Persistent Sessions

VTAM MultiNode Persistent Sessions (MNPS) provide non-disruptive session recovery in case of a planned or unplanned outage of an application, VTAM, or even a whole system. Like VTAM GR, MNPS uses a CF structure to hold data (in this case, session data) that may be required on a different node should a VTAM application using this capability be restarted on another VTAM node.

The default name of the MNPS CF structure is ISTMNPS. Like VTAM GR, if this structure is defined, VTAM will by default try to connect to the structure unless you specify STRMNPS=NONE or STRMNPS=some_other_structure_name in the VTAM start options.

Restriction: While VTAM provides the ability to specify a structure name other than ISTMNPS, it is not recommended to use a different structure name on some systems in the sysplex in an attempt to create subplexes.

This capability was provided to give customers the ability to have a test VTAM MNPS setup active within the same sysplex as a production VTAM MNPS group. While this could be viewed as providing a subplex capability (some systems could use one VTAM MNPS structure and others could use another), this is not the intended purpose of this capability.

If the “incoming system” is actually a set of systems that currently use VTAM MNPS, and the existing systems in the target sysplex are also using VTAM MNPS, and your target environment is a BronzePlex (meaning that you don’t want *all* the VTAMs in the sysplex to communicate with each other), then you again have a problem. While it is technically possible to have two different “VTAM MNPSplexes” in the sysplex, this is not recommended for production—meaning that one of the VTAMplexes is going to have to discontinue its use of VTAM MNPS.

11.2 One or more VTAMplexes

We define a VTAMplex to be the set of z/OS systems whose VTAMs are connected to each other, thus enabling an LU on any of those systems to be able to log on to a VTAM application on the same or any other system in the VTAMplex. While it is possible to connect systems using mechanisms other than XCF, for the purpose of this discussion, we are going to say that to be a member of the VTAMplex, a VTAM must be connected to the ISTXCF group.

When merging a system into the sysplex, you need to decide whether that system will be in the same VTAMplex as the other systems, or if you want to keep it separate.

In a BronzePlex, you would normally be aiming to have as little sharing as possible. If this is your target environment, it is likely that you will not want to give LUs on the incoming system access to applications on the target sysplex and vice versa.

While this is a valid environment, you should be aware that if you do not want to connect the VTAMs, then you must do *one* of the following:

- Stop VTAM on the incoming system(s) from joining the ISTXCF group
- Stop all VTAMs on the target sysplex systems from joining that group

Remember, as well, that by stopping a system from joining ISTXCF, you are limiting the sysplex-related benefits that TCP on that system can use. This is discussed in more detail in Chapter 12, “TCP/IP considerations” on page 175.

In a GoldPlex or a PlatinumPlex, it is more likely that you would want to provide the capability for LUs on any system in the sysplex to be able to logon to VTAM applications on any system in the sysplex. In this case, you probably would allow VTAM to connect to the ISTXCF group.

Be aware, however, that simply placing all the systems in the same VTAMplex will not, by itself, provide you with improved application availability or workload balancing. Being in a single VTAMplex provides the enabling infrastructure; however, you still have to design and deploy your applications in a manner that allows you to exploit the capabilities.

11.3 Considerations for merging VTAMplexes

Table 11-1 contains in checklist format a list of considerations, should you decide to merge two or more systems that are using VTAM.

Table 11-1 Considerations for merging VTAMplexes

Consideration	Note	Type	Done?
Neither VTAM nor TCP/IP currently support sub-clustering using XCF in a sysplex environment	**1**	**B**	
By default, all the APPN-capable VTAMs in a sysplex will communicate with each other unless you explicitly specify otherwise	**2**	**B, G, P**	
TCP/IP is reliant on VTAM for access to XCF services	**3**	**B, G, P**	
If possible, move to a common network id (NETID) if you wish to have VTAM-to-VTAM connection between the incoming systems and the target sysplex systems	**4**	**G, P**	

The “Type” specified in Table 11-1 relates to the sysplex target environment—**B** represents a BronzePlex, **G** represents a GoldPlex, and **P** represents a PlatinumPlex.

Notes indicated in Table 11-1 refer to the following points:

1. Neither VTAM nor TCP/IP currently support sub-clustering. That is, you cannot have all VTAMs using dynamic XCF links, but say that systems A, B, and C can only communicate with each other, and systems D and E can only communicate with each other.

   If a VTAM connects to the ISTXCF group, you have no way of stopping it from communicating with all the other VTAMs in the sysplex that are connected to the same group. And because you have no control over the name of the group, all VTAMs that connect to XCF will connect to the same group and therefore be able to communicate with each other.

   This is of particular note if the incoming system (which may actually comprise a sysplex environment already) and the target sysplex network environments both already use dynamic XCF links—as there is currently no way of letting this continue unless you are willing to let all the VTAMs connect to each other.

2. By default, all the APPN-capable VTAMs in a sysplex will communicate with each other. If you want to maintain separation between the incoming system(s) and the target sysplex networks, then you must specify XCFINIT=NO for all the VTAMs in one of the subplexes.

   For more information on how VTAM uses XCF, and how you can control this, refer to 11.1.4, “How VTAM uses XCF” on page 170.

   An alternative would be to define connectivity using MPC+ via AHHC (available in CS/390 R3 and above), which has the benefits of better throughput and less CPU overhead. However, VTAM definitions are required in order to set up and maintain VTAM-to-VTAM connectivity via MPC+.

   The reason you might do this is for granularity. With XCFINIT=YES, dynamic XCF links are established to *every* other system in the sysplex that has also specified (or defaulted to) XCFINIT=YES, regardless of NETID affiliation. With MPCs, you only get MPC links to the nodes that you define them to.

   Therefore, you could set up two subplexes in the same sysplex and allow each subplex to create “meshed MPCs” between their own hosts without forcing them to also establish links to the hosts that belong to the other subplex. However, note that this granularity is at the *host* level, not at the APPL level. That is, you cannot use this capability to provide access to just a subset of applications in a host—either all the applications on the host are available to all connected systems, or none of them are.

3. In order for TCP/IP to be able to use the full range of its sysplex-specific functions, the corresponding VTAMs must be connected to the ISTXCF group.

4. Although we typically recommend that all of the VTAMs in a sysplex share the same network-id, this is not a requirement. Dynamic XCF links can be used between Network Nodes (NNs) and End Nodes (ENs) that have different NETIDs. You can also use Dynamic XCF links between NNs with different NETIDs (but with BN=YES specified to enable Border Node). In this case, you can establish Dynamic XCF links between these EBNs.

   However, VTAM MNPS and VTAM GR will not work unless *all* nodes that participate in the function (such as “application owners” or, for GR, “Resource Selector Nodes”) have the same NETID.

11.4 Documentation

The following documents contain information that may be useful to you when deciding how to handle your VTAMplexes:

- *SNA in a Parallel Sysplex Environment,* SG24-2113
- *TCP/IP in a Sysplex*, SG24-5235
- *z/OS Communications Server: SNA Network Implementation Guide,* SC31-8777

12

TCP/IP considerations

This chapter discusses the following aspects of moving a system that is using TCP/IP into a sysplex where one or more of the systems are also using TCP/IP:

- Is it necessary to merge TCP/IP environments, or is it possible to have more than one TCP/IP environment in a single sysplex?
- A checklist of things to consider should you decide to merge the TCPs.
- A list of documentation that can assist with this exercise.

12.1 One or more TCPplexes

TCP/IP uses MVS XCF messaging to provide a number of sysplex-unique capabilities:

- When TCP/IP connects to the EZBTCPCS XCF group, XCF notifies every other TCP/IP in the sysplex about the new instance. As a result, it is not necessary to explicitly define the stacks within the sysplex to each other.

  Note that even though the stacks are aware of each other's existence, they cannot communicate with each other unless you either enable Dynamic XCF on the stacks or define explicit connections between the stacks.

- TCP/IP can optionally use XCF for carrying IP traffic between the stacks in the sysplex. It uses the ISTXCF XCF group for this transport. This capability is enabled if you specify DYNAMICXCF in the TCP/IP parameters.

  You can also define a static XCF link between TCPs in two different LPARs by using an MPCPTP Device and Link statement. This uses the same XCF group for IP packet transport (ISTXCF), but does not require Dynamic XCF.

  You cannot use a static XCF link for Sysplex Distributor, or for non-disruptive movement of Dynamic VIPAs, but it can be used for pure IP connectivity over XCF links when Dynamic XCF is not needed or not desired.

- TCP/IP uses MVS XCF Messaging to support Dynamic Virtual IP Addressing (DVIPAs). Dynamic XCF is only required for non-disruptive VIPA movement and for Sysplex Distributor. Application DVIPAs (see 12.2.3, "Dynamic VIPA" on page 181) can be activated without Dynamic XCF, as long as they are always active on only one stack at a time.

Therefore, there could be two definitions of a TCPplex:

1. The first one is the set of TCP/IP stacks that connect to the EZBTCPCS XCF group, and therefore are aware of each other's existence.

   All of the TCPs in a sysplex automatically connect to the EZBTCPCS XCF group, and therefore any LPAR which has z/OS and an active TCP/IP could be considered part of the "TCPPlex". Since just about every z/OS LPAR will have an active TCP/IP, the scope of the TCPplex would generally be the same as the sysplex. Because you have no control over the name of the XCF group, nor whether TCP/IP connects to it or not, it is not possible to have more than one of this type of TCPplex in a sysplex.

   However, the fact that the TCP/IP stacks are aware of each other does not of itself mean that a user on one system can connect to a TCP/IP application on another system in the sysplex. Unless you specifically connect the stacks (through a feature like Dynamic XCF), the fact that the TCP/IP stacks are connected to the EZBTCPCS XCF group is transparent and has no impact on users of the systems. Therefore, in the context of this chapter, we will *not* use this definition of a TCPplex.

2. The other type of TCPplex is the set of TCP/IP stacks that can communicate with each other and send IP traffic to each other. The link between the stacks may be provided by static links that you define, or by you specifying that the stacks should use Dynamic XCF.

   Unless you use IPSec filter rules to stop it, anyone on a stack that has specified DYNAMICXCF can communicate to a TCP/IP application on any other stack in the sysplex that has also specified DYNAMICXCF. Additionally, because this is a binary function (it is either enabled or it is not enabled), you *cannot* have multiple TCPplex subplexes in a sysplex.

So for example, if the sysplex consists of systems SYSA, SYSB, SYSC, and SYSD, and you want to use Dynamic XCF to provide TCP/IP communication between SYSA and SYSB, and between SYSC and SYSD—but you *don't* want SYSA or SYSB to be able to communicate with SYSC or SYSD—that is *not* possible.

Because specifying DYNAMICXCF *does* make a material difference to users on systems that use this facility, this is the definition of TCPplex that we will use in the remainder of this chapter.

When operating in a sysplex environment, TCP/IP can provide the ability for a collection of z/OS systems to cooperate with each other to provide a single-system image for clustered IP server applications. The location of the IP server applications within the TCPplex will be transparent to the end user. To them, it appears there is a single instance of the application. They are unaware that many instances of the same application are running on multiple systems in the sysplex. This single-system image allows you to dynamically balance workload across the sysplex and minimize the effects of hardware, software, and application failures on your end users.

If you fully exploit the functionality of a TCPplex, you will be able to provide your clustered IP server applications with:

- A single IP address for connections to applications in the sysplex cluster
- Workload balancing across multiple systems in the sysplex cluster
- Improved availability
- Improved scalability

12.1.1 Target environments

The main benefit of Dynamic XCF is being able to provide a single-system image for your clustered IP server applications, so that you can deliver dynamic workload balancing across multiple systems, and improve availability by insulating your end users from the effects of hardware, software and application failure.

This can only be achieved in a PlatinumPlex, where the incoming system will be sharing everything with all the other members of the sysplex. This sharing of user data and system resources will enable you to:

- Clone some or all of your existing IP server applications from the incoming system so that they can run on both the incoming system and a number of other systems in the target sysplex
- Clone some or all of your existing IP server applications from the target sysplex so that they can also run on the incoming system

Once your IP server applications from the incoming system are capable of running across multiple systems in the merged sysplex, you will be able to utilize functions such as Dynamic XCF, and therefore Dynamic VIPA and Sysplex Distributor, to provide a single-system image for your clustered IP server applications, with all the availability and workload balancing benefits this implies.

If you are implementing a BronzePlex, you will be sharing a bare minimum of system resources and you won't be sharing any user data. This lack of data sharing will prevent you from migrating your IP server applications to other systems in the sysplex. You also will not want users on the incoming system network connecting to applications on the target sysplex sysplex, nor users on the target sysplex network connecting to applications on the incoming system.

In this case, there should be no changes to TCP/IP in either environment (because, from a TCP/IP perspective, nothing will have changed as a result of the merge).

Note: This assumes that the incoming system does not already use Dynamic XCF.

If you are implementing a PlatinumPlex, and will be exploiting the TCP/IP sysplex features as described above, then you *must* have a TCP/IP connection between the incoming system and the target sysplex. The normal way to do this would be to specify DYNAMICXCF on every system. If you want to exploit features like Sysplex Distributor, you *must* specify DYNAMICXCF on all systems that will be in the TCPplex. If TCP/IP on the target sysplex and the incoming system will be connected, then you must review the TCP/IP network definitions—this is discussed in greater detail in 12.2.2, “Routing in a sysplex” on page 179.

If you are implementing a GoldPlex, the TCPplex considerations depend on whether you wish to connect the TCP/IP stacks on the incoming system and target sysplex systems:

- If you do *not* wish to make this connection, then the considerations are exactly the same as for a BronzePlex.
- If you *do* wish to make this connection, then the considerations are identical to those for implementing a PlatinumPlex.

In summary, from a TCP/IP perspective, there are really only two target environments—one where all the TCPs are connected and using Dynamic XCF, and one where the incoming system TCP/IP is not connected to the target sysplex TCPs; note the following:

- If all the TCPs *are* connected and using Dynamic XCF, the incoming system can exploit all the sysplex TCP/IP capabilities.
- If all the TCPs are *not* connected, then *either* the target sysplex *or* the incoming system can use all the TCP/IP sysplex features—but not both of them.

12.2 TCP/IP in a sysplex

TCP/IP provides a number of sysplex-specific functions that you can use to provide your clustered IP server applications with a single-system image, so that you can deliver improved availability and dynamic workload balancing capabilities. For the incoming system to take advantage of many of these functions, all the TCPIP stacks must be connected, and the IP server applications from the incoming system will need to be deployed onto some of the other systems in the merged sysplex. These functions are discussed in the following sections.

12.2.1 DYNAMICXCF

When you specify DYNAMICXCF, TCP/IP uses MVS XCF messaging to dynamically discover the other TCP/IP stacks in the sysplex. It will then automatically define the XCF links that are used to communicate between the TCP/IP stacks in the sysplex. This feature simplifies configuring the communications links between TCP/IP stacks in the sysplex, because it removes the need to define the links manually.

These links are used to direct IP packets between the TCP/IP stacks, and to exchange information about the IP addresses that are supported by each TCP/IP stack. Information is exchanged whenever:

- A new TCP/IP stack is added.
- A TCP/IP stack is halted or fails.
- An IP address is added or deleted.

This information exchange ensures that each TCP/IP stack in the sysplex is aware of all the active IP addresses in stacks in the sysplex. As changes occur to the IP configuration, this dynamic messaging maintains the currency of the sysplex IP address table in each TCP/IP stack. This information exchange contributes to the TCPplex being able to provide a single-system image for clustered IP applications.

In a TCPplex, each TCP/IP stack knows about all the active IP addresses in the sysplex. This raises the following issues when merging the network of the incoming system into the target sysplex:

- Differences in the IP network addressing scheme used in both the incoming system and the target sysplex must be resolved. If creating a PlatinumPlex, it is probable that the IP addressing of the incoming system will have to be changed, with associated changes for the DNS or client configurations.
- User applications on the incoming and target systems may have been written to use the same port numbers, which may restrict use of Dynamic VIPAs in the sysplex prior to the installation of z/OS 1.4.
- All the active IP addresses are known to all the systems in the TCPplex. This may not be a desirable situation.

  For example, you may be an outsourcer, where you have two competing customers sharing a sysplex, but you don't want them to see each other's IP applications. You may need to provide some form of network isolation, in which case you will probably *not* use Dynamic XCF and not connect the TCPs.

Dynamic XCF uses an XCF group called ISTXCF for communication between TCP/IP stacks. This group is also used by VTAM for dynamically defining VTAM-to-VTAM connections. The group name is fixed at ISTXCF and cannot be customized. This is the reason there cannot be more than one TCPplex in a sysplex—every TCP/IP stack that uses Dynamic XCF connects to the same group and can therefore communicate with every other TCP/IP stack that is also using Dynamic XCF.

Dynamic XCF is implemented by specifying the DYNAMICXCF statement in the IPCONFIG. This automatic definition of IP links removes the need to manually define or update links whenever a new TCP/IP stack is added to the TCPplex.

The pre-requisites for Dynamic XCF are:

- There must be a common IP addressing scheme across all TCP/IP stacks that will use this support.
- Dynamic definition of VTAM-to-VTAM connections must be enabled:
  - VTAM must be executing in an APPN environment.
  - The APPN node must support HPR using RTP.
  - As long as these conditions are met, VTAM will connect to XCF by default.

12.2.2 Routing in a sysplex

The ability to route packets across networks is a fundamental part of the TCP/IP architecture. Most TCP/IP networks are made up of many separate physical networks linked together. The TCP/IP network addressing scheme defines network and host addresses and the function of routers is to allow hosts on one network to connect to hosts on another network. Routers are TCP/IP hosts with multiple network connections with the ability to route packets from one TCP/IP network to another. Even within one organization, most TCP/IP networks will consist of many different networks connected by routers.

It is well beyond the scope of this chapter to give an overview of TCP/IP routing, but it is important to discuss briefly the differences between static routing and dynamic routing:

- With *static routing*, paths to different networks are hardcoded in tables held on each TCP/IP host. Static routing is sufficient for small stable networks, but as network size and complexity increases, it becomes impossible to administer the required entries across all routers in the network.

  Note: Static routing should *not* be used if you wish to avail of any of the advanced TCP/IP availability functions discussed in this chapter.

- With *dynamic routing*, paths to different networks are learned and updated dynamically by routers on the network. While requiring greater effort to implement initially, dynamic routing removes administrative overhead and allows much better recovery from hardware failures provided there is redundancy in the network design. Most of the functions we will discuss which provide improved availability or workload balancing in a sysplex rely on a dynamic routing protocol being enabled in the sysplex and attached networks.

  There are two dynamic routing protocols that are likely to be used within organizations: RIP and OSPF.

  - RIP, or Routing Information Protocol, is a long-established internal gateway protocol (IGP) designed to manage relatively small networks. It has many limitations, some of which are removed by a later version of RIP called RIP Version 2 (often referred to as RIP 2 or RIP V2).

    RIP is the most widely supported protocol because it has been around the longest. Communications Server for z/OS supports RIP V2, and it can be implemented using the OMPROUTE server.

  - OSPF, or Open Short Path First, is a newer IGP that was designed for large networks. It removes many of the limitations found in RIP and tends to be the routing protocol of choice within large organizations. Among the benefits of OSPF is the improved time to route around network failures, and the reduced network traffic required to maintain routing tables when compared to RIP.

    OSPF is not as widely supported as RIP, but this is changing rapidly. The current version of OSPF (V2) is described in RFC 2328 and is implemented by Communications Server for z/OS by the OMPROUTE server.

When merging TCP/IP networks, a choice has to made as to which routing protocols to use. This decision cannot be taken by the z/OS network system programmers in isolation. The routing protocols implemented by Communications Server in the sysplex must match those used by the rest of the TCP/IP network to which the sysplex is attached. If TCP/IP networks are being merged, then there will usually have to be a TCP/IP network redesign and one routing protocol will be chosen, probably OSPF.

Important: It is possible to run both RIP and OSPF at the same time, but Communications Server will always prefer the OSPF route to the RIP route to the same destination.

When merging networks, it may be required to run OSPF over some network interfaces and RIP over others.

For a full discussion of TCP/IP routing, refer to the IBM Redbooks *TCP/IP Tutorial and Technical Overview*, GG24-3376, and *TCP/IP in a Sysplex*, SG24-5235.

12.2.3 Dynamic VIPA

A Virtual IP Address (VIPA) is an availability feature. VIPA was developed to make IP addresses independent of hardware interfaces. Prior to VIPA, IP addresses were assigned to specific physical network interfaces. If the network interface failed, connectivity to the server application was disrupted.

VIPA provides an IP address that is supported by a TCP/IP stack instead of being assigned to a specific network adaptor. If the TCP/IP stack supporting a VIPA address is connected to the network by a number of network interfaces, then the failure of a single network interface will not interrupt connectivity to the server application. VIPA requires that dynamic routing using RIP or OSPF is enabled in the network.

Dynamic VIPA takes the VIPA concept a step further. Dynamic VIPA allows the IP addresses to move from one TCP/IP stack to another. The original or Static VIPA was largely connected with one TCP/IP stack. If the system, TCP/IP stack, or server application fails, then Dynamic VIPA allows the VIPA to be moved to another TCP/IP stack either manually or by some automation. (This assumes there is another instance of the server application on another system ready to take over from the failing instance.)

There are two types of Dynamic VIPA:

- Application Dynamic VIPA, where the virtual address is associated with a particular instance of an application. An Application DVIPA is only active in one stack in the sysplex. There are various ways to associate an Application DVIPA with an application:
  - The application may be written specifically to do so.
  - The application may be associated with the DVIPA by use of BIND keyword on the PORT definition.
  - By use of the MODVIPA utility.

  The Application DVIPA will only exist while the application is running. If the application fails, the application can be restarted in another stack and the DVIPA will be activated by the application. Application DVIPAs are defined to TCP/IP using the VIPARANGE statement in the TCP/IP profile.
- System-Managed Dynamic VIPAs, otherwise referred to as automatic VIPA takeover/takeback. In this instance, the DVIPA is associated with a TCP/IP stack in the sysplex. In the event of a system or stack failure, the DVIPA will be automatically moved to a backup stack without operator intervention. The DVIPA can be defined to move back to the original stack once it is restarted.

  System-Managed DVIPAs are defined using the VIPADEFINE and VIPABACKUP statements in the TCP/IP profile. Communication Server in OS/390 2.10 introduced the Sysplex Distributor function, which uses a particular form of System-Managed DVIPA that is defined by the VIPADISTRBUTE statement. This is discussed in 12.2.6, “Sysplex Distributor” on page 184.

It is possible for a DVIPA to exist on more than one stack at a time, but the TCP/IP stacks will cooperate among each other to ensure that only one stack advertises the DVIPA. The rules for determining which stack advertises the DVIPA are quite complex and depend on how the DVIPA has been defined. A DVIPA must be defined as MOVEABLE in order for TCP/IP to allow it to be activated on a different stack.

The behavior of Application DVIPAs and System-Managed DVIPAs is different:

- Application DVIPAs can be defined as MOVEABLE NONDISRUPTIVE or MOVEABLE DISRUPTIVE. With MOVEABLE NONDISRUPTIVE, an Application DVIPA can be moved back to its original stack in such a way that existing connections stay active to the existing

stack and new connections are directed to the original stack. This provides the least disruption to clients. This behavior will vary according to how the Application DVIPA was activated. For more information, refer to the IBM Redbook *TCP/IP in a Sysplex*, SG24-5235.

- System-Managed DVIPAs can be defined as MOVEABLE IMMEDIATE or MOVEABLE WHENIDLE. If defined as MOVEABLE IMMEDIATE (introduced with OS/390 2.10), then the DVIPAs owned by a failing stack will be taken back immediately by the original stack when it is restarted, even if there are existing connections to the backup stack. Existing connections will be automatically routed to the backup stack, so this process is non-disruptive.

In summary, Dynamic VIPAs, in particular System-Managed DVIPAs, are only likely to be useful as a part of a PlatinumPlex where there is full data sharing and where applications can be run on any system in the sysplex. Dynamic VIPAs do not provide any workload balancing function; they are specifically a high-availability feature.

12.2.4 DNS/WLM - Connection Optimization

Connection Optimization is a function that provides improved availability and load balancing. Connection Optimization has been available for some time, and is no longer considered strategic. While it is still shipped and supported by z/OS, Sysplex Distributor is the strategic solution for connection balancing within the sysplex.

Connection Optimization is a function of Communication Server on z/OS that uses the domain name server (DNS) running on a system in the sysplex to distribute client connections across systems in the sysplex based on WLM performance data.

The clients need to be configured to use a system in the sysplex as their DNS, and Connection Optimization creates a new domain of server applications available in the sysplex. When a client connects to a server, it must use a particular server name which Connection Optimization recognizes as being an application in the sysplex. Connection Optimization then routes the client connection to the system it considers the best one to deal with the new request, based on information provided by WLM.

Connection Optimization requires that all stacks in the sysplex register with WLM by using the SYSPLEXROUTING keyword in the IP configuration profile. This provides Connection Optimization with information on all available IP addresses in the sysplex. Systems in the sysplex must be running in WLM Goal mode. Server applications must also register themselves with WLM—this requires application coding. Communications Server provides Connection Optimization support in TN3270, FTP, and CICS Sockets.

Connection Optimization has a number of limitations that restrict its attractiveness:

- DNS/WLM is only for installations that decide to place a name server in a z/OS sysplex.
- The server applications that use this are probably limited because the applications must be capable of registering with DNS/WLM.
- The clients must use the correct server names (that is, they cannot use the IP address of the server), or they will bypass Connection Optimization.
- Connection Optimization is really only suitable for connections that are of long duration.
- Client TCP/IP stacks tend to cache DNS entries—which means that they do not receive updated DNS information.

If the incoming system is going to be part of this domain and use the DNS/WLM to provide workload balancing, do the following:

- Update the forward domain file with hostnames-to-IP addresses mapping.
- Update the reverse domain or in-addr.arpa file with IP addresses-to-hostnames mapping.
- Note that there may be other config files to update, too!

DNS/WLM is an availability and load balancing tool. This implies a number of cloned applications sharing data executing across multiple servers. This may not apply to the incoming system, particularly in a BronzePlex or a GoldPlex, as there is no data sharing.

12.2.5 External IP workload balancing

There are many solutions for load balancing in a TCP/IP environment that do not use any z/OS functionality. The problem to be solved is how to distribute client connections across a number of server application instances. Most common solutions present a single “cluster” IP address to clients, and then use some mechanism to distribute client connections among target IP addresses. The mechanism may be as simple as a round-robin distribution with the ability to detect servers that are failing to respond.

Ideally, for applications that reside in a sysplex, we would like to base the routing decision on information from WLM. We will discuss two solutions that attempt to use input from WLM running on a z/OS system:

- Network Dispatcher (NDR) is a workload balancing and enhanced availability function supplied as part of the IBM WebSphere Edge Server that runs on NT or UNIX. NDR consists of the following functions:
  - An executor function that load-balances based on the type of connection request received.
  - A manager function that sets the weighting for the target servers.
  - A number of advisor functions that provide input to the manager that influence the weight assigned to individual servers. One of the advisors connects to the WLM port (10007) on z/OS and gets workload information about systems in the sysplex.
- Cisco Multi-Node Load Balancing (MNLB) is a similar function for Cisco routers. MNLB consists of a service manager function that runs on a Cisco router that makes the load balancing decisions, and a forwarding agent function that runs on many Cisco routers.

  When a Cisco router running the forwarding agent receives a client request for a connection to the “cluster” IP address, it will contact the service manger, which returns a selected target IP address. The router will then forward the client connection to that address. As discussed in 12.2.6, “Sysplex Distributor” on page 184, z/OS 1.2 provides support for MNLB in a Sysplex Distributor environment.

External IP workload balancing does not require any sysplex functionality to be enabled in TCP/IP. It does not require the use of Dynamic VIPA or Dynamic XCF links. These solutions, however, do assume that all the target applications are in a single WLMplex. IP workload balancing assumes that there will be a number of application server instances across the sysplex. This is only likely to be the case in a PlatinumPlex, so as with most other high availability or workload balancing solutions we look at, they are not as relevant in a BronzePlex or GoldPlex.

12.2.6 Sysplex Distributor

Sysplex Distributor is a function that provides improved availability and load balancing. It uses an enhancement of Dynamic VIPAs as discussed previously, and was first provided with OS/390 2.10.

Sysplex Distributor provides a single image view of the sysplex to the network. The sysplex presents one IP address that clients can connect to, and then routes client connections to systems in the sysplex based on workload balancing and application availability decisions. Sysplex Distributor can base its workload balancing decisions on data supplied by WLM, and on policies defined in the Sysplex Distributor policy.

Sysplex Distributor uses a Distributed Dynamic VIPA, which means that the DVIPA exists on one or more systems in the sysplex at the same time. Only one system, however, will advertise the DVIPA to the external network. That system is called the *distributing system* and it defines the Distributed DVIPA by means of the VIPADEFINE and VIPADISTRIBUTE keywords in the TCP/IP profile.

The Sysplex Distributor function requires that all participating TCP/IP stacks in the sysplex be defined with the DATAGRAMFWD and DYNAMICXCF keywords. The SYSPLEXROUTING keyword should be specified if WLM-based workload balancing is desired. Dynamic routing using RIP or OSPF is highly desirable for providing backup for the distributing stack, but is not strictly required.

In the event of a failure of the distributing system, one of the other systems in the sysplex can act as backup and can take over the Sysplex Distributor function. That system will then advertise the DVIPA to the external network and accept incoming connections from clients and distribute them around the sysplex. When the original distributing system is restarted, it can take back the distributor function in much the same way that conventional DVIPA takeover/takeback works. This takeover can be non-disruptive to client applications.

Starting with z/OS 1.2, Sysplex Distributor can act as the service manager for Cisco's multi-node load balancer (MNLB). As discussed in 12.2.5, "External IP workload balancing" on page 183, Cisco's MNLB is a load balancing solution whereby a cluster IP address is presented to clients and client connections are distributed among target IP addresses as defined in the Cisco service manager function that runs on a Cisco router.

With z/OS 1.2, the Cisco routers can be defined only as forwarding agents and the service manger function is provided by Sysplex Distributor. This allows the distributing stack to be bypassed for inbound connections, but the choice of target stack will still be based on the same decisions that Sysplex Distributor uses when distributing connections within the sysplex.

Note that Cisco forwarding agents cannot handle IPSec-encrypted (VPN) traffic. The TCP header is encrypted, and the IP addresses in the IP packet header may be different, so the Cisco routers do not have the information they need to bypass the routing (distributing) stack.

Sysplex Distributor provides a high availability and workload balancing solution that removes most disadvantages of the DNS/WLM and external IP workload balancing solutions mentioned previously. Note that Sysplex Distributor is probably only applicable if you are implementing a PlatinumPlex. While it is possible to configure Sysplex Distributor so that client connections are only routed to specific server instances, this would defeat the purpose of it. Sysplex Distributor is most beneficial when all systems share data and where server applications are running on most, if not all, systems in the sysplex.

12.2.7 TCPplex prerequisites

The are no prerequisites or changes of any type required if you are not connecting the incoming system TCP/IP to the target sysplex TCP.

If you *are* connecting the TCPs using Dynamic XCF, the following prerequisites apply:

- All the IP addresses in the merged sysplex network must be unique.
- VTAM must be executing in APPN mode.
- The APPN node must support HPR using RTP.
- Dynamic definition of VTAM-to-VTAM connections must be enabled.

The XCF group ISTXCF is created during VTAM initialization when these requisites are met. If the incoming system does not satisfy these prerequisites, your incoming system will not be able to participate in a TCPplex with the target sysplex systems.

12.3 Considerations for merging TCPplexes

This section contains, in checklist format, a list of things that must be considered should you decide to merge two TCPplexes (that is, you plan to connect two TCP/IP stacks that were not previously connected).

Table 12-1 Considerations for merging TCPplexes

Consideration	Note	Type	Done?
Merging the incoming system into the target sysplex could create visibility across former physical boundaries.	1	B, G, P	
There can be only one TCPplex in the sysplex, because neither VTAM or TCP/IP currently support sub-clustering within a sysplex.	2	B, G, P	
If you plan to exploit DNS/WLM, ensure all your clients on both the incoming system and the target sysplex are using a DNS to resolve the IP addresses of their server applications.	3	B, G, P	
Review the IP addresses used in the client networks on both the incoming system and the target sysplex. If some client IP addresses are duplicated, re-number them or use Network Address Translation to guarantee unique IP addresses, from the sysplex perspective.	4	G, P	
Review any server application port numbers used to make sure there are no duplicates. The server applications could be either user-written, or standard servers like DB2 where it may be required to run multiple instances on a single system.	5	G, P	
Implement the same routing protocol in the incoming system as the target sysplex, either OSPF or RIP.	6	G, P	
Set up IP filtering rules to provide network/IP server application isolation.	7	G, P	
In order to exchange IP packets using MVS XCF messaging, the VTAM interface to XCF must be active.	8	G, P	

Consideration	Note	Type	Done?
Implement system symbols for TCP/IP config files and share files across the systems in the sysplex: ► TCPIP profile and TCPIP.DATA	**9**	**G, P**	

The “Type” specified in Table 12-1 relates to the sysplex target environment—**B** represents a BronzePlex, **G** represents a GoldPlex, and **P** represents a PlatinumPlex.

The notes indicated in Table 12-1 refer to the following points:

1. All TCP/IP stacks in the sysplex will be aware of each other’s existence, even if you don’t use Dynamic XCF. However, IP packets cannot be transferred between stacks unless you provide connectivity via static connections or Dynamic XCF.

   Note: Having all the TCP/IPs being aware of each other should not normally cause any problems or concerns; however, some installations may not like the IP addresses being known across the systems.

2. If TCP/IP is using XCF to transfer packets between stacks in the sysplex, regardless of when Dynamic XCF or static XCF links are being used, TCP/IP uses the XCF group ISTXCF for communication within the sysplex—this name is fixed and cannot be changed.

   You might not want a single TCPplex because it is not possible to separate one IP network from another. All the IP addresses in the TCPplex will be available to all the systems in the TCPplex. This may not be a desirable situation.

   A single TCPplex demands unique IP addresses across the entire TCPplex.

3. You should ensure that your clients are using a DNS to access their IP server applications, rather than explicit IP addresses. The use of DNS makes it easier to change IP addresses, should this be required.

4. IP addresses must be unique when they reach the server. Duplicate IP addresses could occur when merging intranets as a result of using private IP addressing schemes. A solution would be to re-number the addresses, or deploy a NAT function between the clients and servers.

5. A server application will usually listen on a specific port. Most servers have agreed or default port numbers (for example, 23 for TN3270 or 446 for DB2). A client application connects to a socket, which is identified by a combination of the IP address and port number.

   It is acceptable for duplicate port numbers to exist on *different* TCP/IP stacks, but a problem will arise if it is required to run more than one application that uses the same port connected to the same stack. For example, if a decision is made to run two DB2s on one system, then they cannot both listen on port 446 if they bind to the same IP address. Application DVIPAs may provide a workaround in this situation, if port numbers cannot be changed. Sysplex Distributor, which presents one sysplex IP address to client applications, also requires unique port numbers for different applications.

6. You can run both RIP and OSPF, but this makes administration confusing and is not recommended.

7. You could use IP filtering rules to prevent users from one stack accessing server applications that belong to another stack.

8. TCP/IP potentially uses two XCF groups. TCP/IP will automatically connect to the EZBTCPCS group for discovering other stacks in the sysplex, and subsequently for exchanging IP addresses and Dynamic VIPA configuration information. The VTAM interface to XCF does not have to be active for this.

However, if you wish to use XCF for exchanging IP packets between stacks (whether you are using Dynamic or static XCF connections), VTAM must be started and must be set up to start its interface to XCF. This capability requires that VTAM is running in APPN mode.

9. The combination of system symbols and dynamic discovery of TCP/IP stacks makes it easier to add more systems to the sysplex.

12.4 Tools and documentation

The following documents should prove useful should you decide that you want to have all the systems in the same TCPplex:

- *TCP/IP Tutorial and Technical Overview*, GG24-3376
- *TCP/IP in a Sysplex*, SG24-5235
- *z/OS Communications Server: IP Configuration Guide*, SC31-8775

13

RACF considerations

This chapter discusses the following aspects of moving a system that is using its own RACF database into a sysplex that is currently using a shared RACF database:

- Is it necessary to merge RACF databases, or is it possible to have more than one RACFplex in a single sysplex?
- A checklist of things to consider should you decide to merge the databases.
- A methodology for doing the merge.
- A list of tools and documentation that can assist with this exercise.

Note: This chapter does *not* provide a step-by-step guide to merging two RACFplexes. This is a complex subject, and the steps and considerations are different in every installation.

However, we *do* provide a list of considerations to keep in mind when deciding whether to merge RACFplexes, as well as hints and tips based on actual experiences. We strongly recommend working with someone that has worked on at least two such merges in the past should you decide to proceed with the merge.

13.1 One or more RACFplexes

The definition of a RACFplex is the set of systems that share a single set of RACF databases. It *is* possible to have more than one RACFplex in a single sysplex, however, there are a number of restrictions should you decide to implement this configuration.

There are a number of benefits if all the systems in the sysplex are in a single RACFplex. One of the functions available to systems in a RACFplex is command propagation between all the systems in the RACFplex. Therefore, instead of having to issue a given RACF command on every system, you just have to issue it on one system, and RACF will automatically propagate it to the other RACFs in the RACFplex using XCF. The name of the XCF group used for this communication is fixed, so this feature can only be used by one RACFplex per sysplex. This feature is optional, and is enabled by setting the sysplex communication flag in the ICHRDSNT module. The commands that are propagated are:

- RVARY SWITCH
- RVARY ACTIVE
- RVARY INACTIVE
- RVARY DATASHARE
- RVARY NODATASHARE
- SETROPTS RACLIST (classname)
- SETROPTS RACLIST (classname) REFRESH
- SETROPTS NORACLIST (classname)
- SETROPTS GLOBAL (classname)
- SETROPTS GLOBAL (classname) REFRESH
- SETROPTS GENERIC (classname) REFRESH
- SETROPTS WHEN(PROGRAM)
- SETROPTS WHEN(PROGRAM) REFRESH

Another benefit of having all systems in the same RACFplex is that all systems can benefit from placing RACF buffers in the RACF structures in the CF. There can be only one set of RACF structures in a sysplex, and all systems using those structures must be sharing the same RACF database. Using the RACF structures in the CF can provide significant performance benefits for a number of reasons:

- RACF now has a far larger number of buffers accessible with very fast access times. While access to the buffers in the CF is not as fast as the buffers in MVS storage, it is still many times faster than reading from DASD.
- When a buffer gets updated, only that buffer gets invalidated on any other systems that also have a copy of the buffer. Previously, when a buffer is updated, other systems will invalidate *all* their in-storage buffers.
- If RACF use of the structures in the CF has been enabled, RACF will use a completely different serialization design, which uses the CF data to eliminate the need for cross-system serialization for most accesses to the database. This provides a further performance benefit.

There is also a security benefit from having just one RACFplex per sysplex. Assuming that all the resources in the sysplex (programs, DASD data sets, tapes, and so on), are potentially accessible from any system in the sysplex, having a single RACFplex ensures that the resources are protected in the same way, regardless of from which system they are accessed. If there is more than one RACFplex (and therefore, more than one set of RACF databases), it is possible, even likely, that some resources will be protected by different RACF profiles, with different access lists.

A different aspect of this problem is that the sysplex architecture encourages the design and implementation of applications that have multiple instances running on separate systems for availability and scalability. However, when you have multiple RACFplexes in the sysplex:

- You limit your flexibility in deploying those multiple instances, since any that share a workload must run within the same RACFplex.
- You therefore end up (potentially) needing to deploy separate clusters of the same application to handle the different security scopes.
- You end up increasing the chance that you will deploy the applications incorrectly, such that an application running in RACFplex#1 will route work incorrectly, sending it to another instance running in a different RACFplex, say RACFplex#2. In that case, several results may occur:
  - The work element may fail because the user is not defined there, or it may fail because the user is defined but the security definitions are different, or
  - It may not fail because, although the user is defined, the security definitions are incorrect, or
  - It may not fail because a different user with the same ID is defined there, and that userid (different person, remember) legitimately has the authority to perform that work, even though the original user would not have that authority if the work ran in the proper RACFplex.
- Note that one such application is system console support. In a sysplex, it is common for an operator on one system to route commands to another system for execution. If the user definitions or security rules are not common between the systems, such commands may either fail unexpectedly or work (when they should have failed).

Despite all of this, it *is* possible to have more than one RACFplex in a sysplex, should you decide to do so. The main benefit of such an approach is that you avoid the one-time cost of having to merge the RACF databases.

If your target environment is a BronzePlex, it is *not* necessary to merge the RACFplexes. Only the minimum set of shared resources is available to both RACFplexes. Relatively few RACF profiles would need to be maintained to provide a consistent security environment, for those shared resources, across the two RACFplexes.

Similarly, in a GoldPlex, where user DASD is not shared, there is still not a lot of benefit in merging the two RACFplexes. However, if you are moving towards a PlatinumPlex, then the merging of RACFplexes will become an issue. Merging RACFplexes can often take a long time and should be completed before large scale DASD sharing is attempted. So it may be worthwhile starting the RACFplex merge project at this stage.

In the case of a PlatinumPlex, it makes little sense to maintain more than one RACFplex. In this environment most, or all, of the sysplex DASD will be online to all systems. There will probably be a unified catalog structure, and thus single SMS and HSMplexes. Trying to maintain two security environments, and yet ensure that all resources are protected in the same manner is, at best, problematic, and at worst, a potential security exposure. For a PlatinumPlex it is *strongly* recommended to merge into a single RACFplexes.

13.2 Considerations for merging RACFplexes

This section contains, in checklist format, a list of things that must be considered should you decide to merge two or more RACFplexes. We say RACFplexes rather than RACF databases, because there are more things to be considered than simply the contents of the RACF databases.

Table 13-1 Considerations for merging RACFplexes

Consideration	Note	Type	Done?
All systems must have physical access to the RACF databases.		**P**	
All systems should have access to CFs that will contain the RACF structures, if you are using that capability.	**1**	**P**	
RACF database range table (ICHRRNG) must be identical on all systems.		**P**	
Check that the RACF database is large enough to contain all profiles.	**2**	**P**	
RACF data set name table (ICHRDSNT) must be compatible on all systems.	**3**	**P**	
RACF class descriptor table (ICHRRCDE) must be compatible on all systems.	**3**	**P**	
RACF router table (ICHRFR01) must be compatible on all systems.	**3**	**P**	
Replace the RACF started task table (ICHRIN03) with **STARTED** class profiles.	**4**	**P**	
The RACF Global Access Checking table should be identical on all systems.	**5**	**P**	
The same RACF classes should be enabled on all systems.	**5**	**P**	
The RACF options in effect should be identical on all systems.	**5**	**P**	
RACF exits should be identical on all systems.	**5**	**P**	
The RACF database templates must match the highest level of RACF.	**6**	**P**	
Check if SECDATA and/or SECLABEL classes are active in any of the systems.	**7**	**P**	

The “Type” specified in Table 13-1 relates to the sysplex target environment—**B** represents a BronzePlex, **G** represents a GoldPlex, and **P** represents a PlatinumPlex. Notes indicated in Table 13-1 are described below:

1. All systems must have access to the CFs containing the RACF structures if you plan on placing the RACF buffers in CF structures. This should not be an issue as in a PlatinumPlex, all systems in the sysplex would normally have access to all CFs anyway.
2. Review the size of the RACF database to be used for the single RACFplex. You can determine how much free space there is in your databases using the RACF database verification utility program (IRRUT200). If there is insufficient free space within the RACF database, then the database can be extended using the Split/Merge/Extend utility program (IRRUT400).
3. If RACF sysplex communication is enabled, when RACF is being initialized, it will compare its data set name table with the data set name table in use on other systems in the sysplex. If any values other than the number of resident data blocks are different, RACF will override those values with the values already in use on the other systems in the sysplex. RACF does not check to ensure that the class descriptor table or the router table are the same on every system, however, for operational simplicity, we strongly recommend using the same version of these three tables on all systems in the sysplex.
4. Though not a requirement, you should replace the ICHRIN03 table with STARTED class profiles. This removes the need to synchronize and maintain ICHRIN03, and can be updated without an IPL.

5. The same RACF classes should be enabled on all systems in the RACFplex, and the RACF options and exits should be identical on all systems. This should be done to ensure resources on all sharing systems are protected in the same manner. To check the options, issue a SETR LIST command on both the target sysplex and the incoming system—before you move the incoming system into the target sysplex, the response to this command should be identical on all systems.

   After you have synchronized the options, check that the RACF Global Access Table is the same on all systems. In particular, you need to be sure Enhanced Generic Naming (EGN) is either enabled on all systems or disabled on all systems before you check the Global Access Table.

6. The RACF database template level must be equal to, or higher than, the highest RACF software level sharing the database. The IRRMIN00 utility is used to update the database templates when a new release or service level is available. The templates are downwardly compatible, meaning that you can run old levels of RACF with a database that has been formatted for a newer release.

7. Security categories, security levels, and security labels require special handling. This is discussed in 13.3.3, “Synchronize RACF options” on page 198.

13.3 Methodology for merging RACFplexes

In this section we discuss the approaches that can be taken to merging RACFplexes, and give some advice about the approach that provides the smallest risk or is the easiest to implement.

The merging of RACFplexes is the unification of two security policies and two user/group registries.

If the same set of security conventions, naming conventions, and so on are in use in both RACFplexes, this may well be a reasonably straight-forward exercise. If, on the other hand, the security policies are significantly different, then this needs to be addressed first. A common set of standards would need to be developed. The merge process could then be used to “standardize” the two security environments.

Similarly, it is possible that the same RACF userid may exist in both security databases, but actually represent two different people. There is also the possibility that a userid in one database could have the same name as a group in the other database. Therefore, the merge process consists not just of checking for duplicate userids and groups, but also checking the actual owner of each userid.

There is no IBM-supported RACF utility to merge two RACF databases. Development of such a utility would be extremely difficult and problematic. Insertion of a single generic profile, for example, could change the access privileges for thousands of data sets. The approach usually taken is to use programs to perform those parts that can be done safely, where there are no decisions to be made, or where a decision can be made unambiguously, for example, if a RACF group exists in one database but not in the other, it can be safely added. There are vendor products that claim to reduce this work significantly, however, you would need to review them specifically in relation to your environment.

In cases where a decision is difficult to automate, then the approach should be to identify that a decision is needed, and detail what is known of the options available. An example would be where a user resides in both RACF databases, but with different attributes. A program could provide the user id and both sets of attributes. The discrepancy can then be resolved

manually. This tends to be an iterative process. With large RACF databases, the discrepancies may be many and take considerable time to resolve. There is an IBM-supplied tool, DBSYNC, that can help you synchronize profiles; this tool is discussed in more detail in 13.3.7, “RACF database merge” on page 199.

Because of the gradual, iterative, approach to synchronizing the two databases, fallback from any change along the way should be quick and simple.

There is no single utility to do the bulk of the work involved. Rather, there are a few utilities which can help, and these will need to be supplemented by additional user-developed utilities. As each installation is unique, different issues will need to be addressed and various utilities developed to aid the merge process.

Important: Do *not* use the RACF split/merge utility (IRRUT400) to merge two RACF databases. It is not a “merge” utility for disparate databases. It is used only to merge databases that have previously been split by the same utility under the control of a range table (for example, recombining a RACF database that has previously been split into key ranges).

Performing a RACF merge requires access to RACF data in a form that is easy to analyze and process. This can be done using an IBM-provided utility such as the RACF database unload utility, IRRDBU00. Data unloaded by IRRDBU00 can be loaded into DB2 tables for easier analysis; this process is described in *z/OS Security Server RACF Security Administrator’s Guide,* SA22-7683.

The RACF command set is rich enough to allow the manipulation of most of the RACF profile fields. Together with the RACF database unload utility, these two features of RACF allow the speedy development of “home grown” utilities to automate large parts of the merge process.

What follows is a step-by-step approach to merging two RACF databases detailing the obstacles that need to be overcome to achieve a single RACF database.

13.3.1 Before you start

If you are going to merge two databases, the fewer entries there are in each database, the easier the merge should be. Therefore, one of the first things you should do to prepare for such a project is to clean up the two RACF databases as much as possible.

Generic or discrete profiles?

RACF supports both generic and discrete profiles. Each type of profile has benefits—it is not correct to say that generic profiles are always necessarily better than discrete profiles. The following points should clarify the situation:

- Generic profiles may provide better performance for DATASET class profiles, and for non-RACLISTed general resource classes. For RACLISTed general resource classes, discrete profiles may in some cases perform much faster during authorization checking.
- Even for DATASET class profiles, if you need to protect one data set uniquely, your choice becomes using a discrete or using a fully-qualified generic. Using a discrete will probably perform better, especially if you end up with a lot of generics for a particular HLQ because you created a lot of fully-qualified generics.
- Discrete profiles are *not* non-strategic. Rather, *appropriate* use of generics (especially for the DATASET class) *is* strategic.
- The key is to have a small number of generics protecting a large number of resources. This gives you good performance characteristics and good administrative characteristics. A secondary key is to have a good data set naming convention, with meaningful HLQs at the application or user or group level, rather than T.something for test and P.something for production, and so on.

A good place to start is by removing all non-existent userids and groups from profiles. Search for userids that have not been used in a long time and delete them if possible. If there are any RACF groups that are not high level qualifiers, are not superior groups for other groups, and have no users connected to them, delete those groups.

Tip: The RACF Remove ID utility (IRRRID00) can be used to remove all references to group IDs and user IDs that no longer exist in the RACF database. Also, you can optionally specify a replacement ID for those IDs that will be removed.

See *z/OS Security Server RACF Security Administrator's Guide,* SA22-7683 for details.

Rationalize data set class and general resource class profiles. RACF profiles are often added and rarely deleted. Over time, profiles accumulate. Clever use of generic profiles and wild card characters can considerably simplify profile trees. Merging of generic profiles is probably the biggest single issue of any merge project, so a thorough cleanup prior to the merge will greatly simplify the merge.

Remove profiles for inactive general resource classes. The REXX exec in Example 13-1 will list all general resources classes that contain profiles in the RACF database. Delete all profiles for classes that are not active.

Note: The report generated by the program in Example 13-1 lists *all* general resource classes that have profiles. This does not mean that the class is actually *active*. To determine if the class is active, use the **SETR LIST** command.

Example 13-1 Sample REXX exec to generate a list of general resource class profiles

```
/* REXX */
"alloc dd(indd) da(racf.unl12) shr reuse" /* alloc IRRDBU00 unload DS */
eof = 0                                  /* clear end-of-file flag      */
hwm = 0                                  /* clear class array ptr       */
do until eof                             /* loop thru IRRDBU00 unload ds*/
```

```
  "execio 100 diskr indd (stem indd."  /* Get some records            */
  if rc = 2 then eof = 1               /* If EOF then set EOF flag    */
  do i = 1 to indd.0                   /* loop thru records read      */
    rtype =  substr(indd.i,1,4)        /* get record type             */
    if rtype = '0500' then             /* is it a genreal resource?   */
      do                               /* yes                         */
        class = substr(indd.i,253,8)   /* get resource class          */
        found = 0                      /* clear flag                  */
        j = 0                          /* clear array subscript       */
        do while j < hwm               /* loop thru class array       */
          if classes.hwm = class      /* have we seen class before ? */
          then found = 1               /* Yes. set flag               */
          j = j + 1                    /* Increment subscript         */
        end                            /*                             */
        if  ¬found then                /* If new class                */
          do                           /*                             */
            hwm = hwm + 1              /* then add to class array     */
            classes.hwm = class        /*                             */
            say class                  /* display each new class      */
          end                          /*                             */
      end                              /*                             */
  end                                  /*                             */
end
```

Tip: You cannot necessarily deactivate a class just because there are no profiles defined in that class. You must decide on a class-by-class basis whether to leave it active, as some classes act as switches rather than as containers for profiles. It may be safe to deactivate any class that doesn't have profiles and that is not listed as PROFDEF=NO in the customer or IBM CDT.

In SYS1.SAMPLIB, member IRRICE contains some very useful sample utilities that can be used to locate problem areas. One sample report, for example, will identify those high-level qualifiers that have more than a specified number of generic data set profiles defined. These can serve as excellent templates to develop your own reports.

For example, the sample job in Example 13-2 uses ICETOOL and SORT to list all users by their last access date. Inactive userids will have old last access dates, or, if the userid has never been used, no last access date.

Example 13-2 ICETOOL sample to list all users by last access date

```
//UNLOAD   EXEC PGM=IRRDBU00,PARM=NOLOCKINPUT
//SYSPRINT  DD  SYSOUT=*
//INDD1     DD  DISP=SHR,DSN=SYS1.RACFPRIM
//OUTDD     DD  DISP=(,CATLG),DSN=&UNL,SPACE=(CYL,(5,5),RLSE),
//              LRECL=4096,BLKSIZE=0,RECFM=VB
//EXTRACT  EXEC PGM=SORT
//SORTIN    DD  DISP=SHR,DSN=&UNL
//SYSOUT    DD  SYSOUT=*
//SORTOUT   DD  DSN=&TEMP,DISP=(NEW,PASS),
//              SPACE=(CYL,(20,5,0)),UNIT=SYSALLDA
//SYSIN     DD  *
  SORT FIELDS=(118,10,CH,A,109,10,CH,A)
  INCLUDE COND=(5,4,CH,EQ,C'0200')
  OPTION  VLSHRT
/*
//REPORT   EXEC PGM=ICETOOL
//TOOLMSG   DD  SYSOUT=*
```

```
//PRINT     DD  SYSOUT=*
//DFSMSG    DD  SYSOUT=*
//DBUDATA   DD  DISP=(OLD,DELETE),DSN=&TEMP
//TOOLIN    DD  *
 DISPLAY FROM(DBUDATA) LIST(PRINT) -
         PAGE  TITLE('USER LAST ACCESS REPORT') -
         DATE(YMD/) TIME(12:) -
         BLANK -
         ON(10,8,CH) HEADER('USER ID') -
         ON(30,8,CH) HEADER('OWNER') -
         ON(19,10,CH) HEADER('CREATE DATE') -
         ON(118,10,CH) HEADER('LAST USED DATE') -
         ON(109,8,CH) HEADER('LAST USED TIME')
/*
```

The JCL contained in Example 13-2 on page 196 will produce the report shown in Example 13-3.

Example 13-3 Report created from ICETOOL example above

```
- 1 -        USER LAST ACCESS REPORT        02/04/23        07:00:26 pm

USER ID    OWNER      CREATE DATE   LAST USED DATE   LAST USED TIME
--------   --------   -----------   --------------   --------------
MSO$1SIR   KYNEF      2001-09-19    2001-09-24       21:46:47
MS2$1PPT   KYNEF      2001-10-10    2001-10-10       11:49:23
MS2$1SIR   KYNEF      2001-10-10    2001-10-10       11:49:28
WATS       KYNEF      2001-09-26    2001-11-05       01:52:08
RAFTEN     KYNEF      2001-03-27    2001-11-29       09:48:55
MAIDA      KYNEF      2000-11-15    2001-12-05       15:06:48
HSHA       KYNEF      2000-08-16    2001-12-10       15:26:32
MS2$2PPT   KYNEF      2001-10-10    2002-01-10       15:34:27
MS2$2SIR   KYNEF      2001-10-10    2002-01-10       15:34:32
VDPUTTE    KYNEF      2001-02-15    2002-01-17       09:20:00
SHEEHAN    KYNEF      2001-02-15    2002-01-17       09:56:17
I#$#IRLM   SYS1       2000-12-05    2002-02-11       11:11:13
LAWSOD     KYNEF      2000-08-16    2002-02-11       11:44:17
#@OPR1     WELLIE2    2001-01-04    2002-02-13       16:31:12
#@OPR10    WELLIE2    2001-01-04    2002-02-13       16:31:13
```

13.3.2 RACF exits

Identify all RACF exits in use on all the target systems. The data security monitor (DSMON) produces the RACF exits report. This report lists the names of all the installation-defined RACF exit routines and specifies the size of each exit routine module. If the RACF communications vector table (RCVT), which contains the address of each RACF exit-routine module except IRREVX01, indicates that an exit-routine module should exist but the module cannot be loaded, or that the entry address does not correspond with the address specified in the RCVT, DSMON prints an error message. DSMON lists IRREVX01 when at least one active exit routine is defined at the time the report is created.

Where possible, remove any unnecessary exits. RACF has been enhanced substantially over the years and there may be ways to achieve the same result without resorting to an exit. Otherwise, reconcile the differences, if any, in the exit processing. Once the databases are merged, the behavior of any exits should be the same to ensure consistent security across all images. So it's best to remove the differences as early as possible.

13.3.3 Synchronize RACF options

Use the **SETR LIST** command to document the RACF options on each system and resolve any differences. It would be a good idea to issue this command from a batch job on each system, so that you have a record of the response. If you issue the command in foreground TSO, the response is sent to your terminal and is not as easily scrolled or saved as if it was issued in a batch job.

If security levels and/or security categories are in use, then ensure they are defined consistently in both RACF databases. For example, if a SECLEVEL of SECRET is defined in both RACF databases, does it have the same security meaning in both environments? If a SECLEVEL of SECRET in one RACF database is the equivalent of a SECLEVEL of TOPSEC in the other then, during the merge, the ADDSD or RDEFINE will need to be modified to convert SECLEVEL(SECRET) to SECLEVEL(TOPSEC).

If security levels are being added as part of the merge, care is needed. Security levels are referenced by name, but what is actually stored in the RACF profiles that use them is a security level *number.* Ensure that any new security levels defined map correctly into the existing security level hierarchy. For example, if one RACF database had SECLEVELs of NOTSEC/20 and TOPSEC/100, then a new security level that is between NOTSEC and TOPSEC should be defined with a number between 20 and 100.

Similar considerations apply to security *categories.* It is important to understand how the category names in one RACF database map to the other. A category of TOPCAT may be defined in both RACF databases, but with different meanings. In that case, a new category would need to be defined and the commands to add incoming profiles referencing TOPCAT modified to use the new category.

In both cases, it is necessary to understand the security mapping in both RCAF databases, and during the merge do any remapping required to maintain the consistency of the security levels and categories.

Once you have identified the profiles that are missing from the target sysplex RACF database and have resolved any issues, you should define those missing profiles in the target sysplex RACF database.

13.3.4 Synchronize the Class Descriptor Table (ICHRRCDE)

Build a “super set” RACF class descriptor table. For any classes that exist in both tables, ensure that they have been defined identically—that is, they are used by the same applications for the same reasons. If there are duplicate class names, but they are used for different applications, then one of the class names (along with any of the applications that use the class) will need to be changed.

13.3.5 Synchronize the router table (ICHRFR01)

Build a “super set” RACF router table. For any classes that exist in both tables, ensure that they have been defined identically.

13.3.6 Global Access Checking Facility

Once the RACF databases are merged, there will be only one Global Access Checking Facility list for all systems in the RACFplex. Use the **RLIST GLOBAL *** RACF command on the incoming system and one of the target sysplex systems to show the Global Access Checking Facility lists and develop a combined list. Remember that this should only be done once all the systems are using the same EGN or NOEGN setting. Also, check that there are no

"logical duplicates"—that is, that there are not two entries specifying SYS1.** and SYS1.*.**. If there are, remove one of the entries. You also need to check to ensure that the same access is being granted. For example, if one system has A.*/READ and the other has A.*/UPDATE, then you need to decide which profile you wish to use in the merged RACFplex.

13.3.7 RACF database merge

Once the above tasks have been performed, the two RACFplexes will be the same in all things except that the RACF databases are still not completely synchronized. Merging RACF databases involves comparing the two RACF databases and creating a "super set" RACF database. At the completion of the merge, the RACF database chosen to be used by all systems in the sysplex must contain profiles that meet the security requirements of both RACFplex's. Thus, part of the merge process involves adding profiles from one database to the other. The complication that arises is where a profile exists in both RACF databases but the duplicate profiles are not identically defined. Some differences can be ignored, such as creation dates, but the majority of the differences must be resolved.

For all but the smallest RACF databases, performing a merge during a system outage is not feasible. The time taken to resolve the conflicts would be prohibitive. A gradual approach is the preferred method, where the differences can be resolved over a period of time. This minimizes and localizes the changes required and greatly reduces the risk. While problems with incorrectly-added RACF profiles are relatively easy to diagnose and correct, a large number of them at one time would still be very disruptive.

Once all differences have been resolved, one of the RACF databases can be discarded. Some sites update both RACF databases during the synchronization phase, while others update only one and synchronize in that direction only. Either method can be used. The one direction method is a little more complicated in that you need to ignore differences in the other direction, while the two-direction method involves a little more work.

Tip: Merging in one direction means you have a set of systems that may not actually operate as desired after the cutover, since all systems were not using an identical database. Synchronizing in both directions is a low-risk way of verifying, before the cutover, that all systems will function correctly in the merged environment.

A useful tool for merging RACF databases is the DBSYNC tool. DBSYNC and other RACF tools can be downloaded from the RACF Web site at:

`http://www.ibm.com/servers/eserver/zseries/zos/racf/goodies.html`

DBSYNC is a REXX exec that compares two RACF databases that have been unloaded with the RACF database unload utility (IRRDBU00). DBSYNC generates the RACF commands required to convert database "a" into database "b" and vice versa. Thus, if a profile exists in database "a" and not in database "b", DBSYNC will generate the RACF commands to delete the profile from database "a" and add the profile to data set "b". Any profiles found in both RACF databases are matched, and any discrepancies reported.

Though not specifically designed to merge RACF databases, DBSYNC is very useful in generating the RACF commands to bring two RACF databases into line. This can be done by selectively using pieces of the command files produced, to add profiles from one RACF database to the other. Being written in REXX, it is easy to modify and debug, should the need arise, to meet the needs of the specific merge project.

Important: DBSYNC was *not* developed as a tool to synchronize two RACF databases—rather, it helps you make one database the same as the other. In the example above, DBSYNC generates two sets of commands: one set will make database “a” the same as database “b”, adding anything that is in “b” and not in “a” and deleting anything that is in “a” that is not in “b”. The other set does the reverse - this is why you have to selectively use just parts of the output from DBSYNC.

The general approach is to run DBSYNC repeatedly over time, making incremental changes to the RACF databases between each run until all the differences have been resolved. This may take many weeks or months, and of course, as the RACF database is a moving target, profiles are being added and deleted all the time.

Note: The RACF remote support facility (RRSF) can be used during the synchronization process to propagate updates to RACF databases. However, the *z/OS Security Server RACF Security Administrator’s Guide,* SA22-7683, under the heading “Synchronizing Database Profiles”, states:-

> “You can use automatic direction to maintain synchronization of RACF database profiles that are *already* synchronized, but you must synchronize the profiles *before* you activate RRSF functions”

Thus RRSF can be used to *maintain* synchronization of already synchronized profiles, but will not actively synchronize them.

Once the differences have been resolved, an outage is required to IPL the incoming system to get it to use the target sysplex RACF database. During this outage a final run of DBSYNC should be performed and any remaining differences corrected. There are some other tasks required during this outage; we will discuss those later in this chapter.

The first step is to run DBSYNC to generate the list of duplicate, but conflicting, profiles. These profiles will need to be examined and a decision made as to how to resolve the conflict. As the conflicts are being resolved, DBSYNC can be rerun to verify that all is going to plan.

Note: If either or both RACF databases contain profiles in the SECDATA class, the database unloads *must* be pre-processed with the DBSECLV exec, before using the DBSYNC exec.

While resolving conflicting profiles, you can use the commands generated by DBSYNC to merge the two databases. RACF profiles often point to other profiles. For example, a user profile can contain references to RACF groups. These groups must exist when the **ADDUSER** command is executed, otherwise the command will fail. DBSYNC takes this into account when generating its command files. Given the large number of RACF commands that will need to be executed, it is wise to break the task up into manageable pieces. This reduces the complexity of the task and the amount of change made at any one time.

While DBSYNC is used to generate the RACF commands, you can also use small REXX execs to identify which bits of the DBSYNC command files you should use. For example, before you add the RACF group profiles, compare the two RACF database unload files and list all the RACF groups in one file but not the other. This provides a count of how many **ADDGROUP** commands you will need to extract and gives you their group names.

The following steps can be used to merge two RACF databases:

Group profiles

Group profiles can be added first, as they have the fewest dependencies. Though they contain a superior group, the DBSYNC utility creates a dummy group and uses that as the superior group for the **ADDGROUP** commands. After all the **ADDGROUP**'s have been generated, DBSYNC generates **ALTGROUP** commands to set the correct superior group. Once these commands have been extracted and run, DBSYNC can be run again. There shouldn't be any **ADDGROUP** or **ALTGROUP** commands generated in the second set of DBSYNC command files.

User profiles

The same procedure can be done for user profiles.

When DBSYNC compares user profiles, it ignores the password fields. The **ADDUSER** commands generated do *not* specify a password, and so the password will be set to the user's default group and the password is set to expired. We will correct the passwords in 13.3.9, "Password synchronization" on page 207.

Connect profiles

Now that the user and group profiles have been processed, group connection profiles can be attempted. There may exist a data set or general resource profile that contains multiple groups and access levels in their access lists. Connecting a user who is already a member of one of these groups to another of these groups may increase their access to resources covered by this profile.

For example, consider a data set profile of SYS1.** with an access list containing GROUP1 with UPDATE access and GROUP2 with READ access. If a user is already connected to group "GROUP2" and as part of the merge we were to connect him to "GROUP1", his access would be upgraded from READ to UPDATE. When connected to multiple groups, the *higher* access is assigned. This may not be a problem, but it may give some users access to resources that may not be desired.

The sample REXX exec shown in Example 13-4 will list all data set profiles that contain more than one group with differing levels of access. This could be used to provide a basis for ensuring that no unwanted access changes occur while adding connect profiles.

Example 13-4 Sample REXX to list data set profiles with a mix of access levels

```
/* REXX */
/*********************************************************************/
/*  List any data set profile that contains an access list with       */
/*  multiple groups and access levels.                                */
/*********************************************************************/
"alloc dd(indd) da(racf.unl13) shr reuse" /* alloc IRRDBU00 unload DS */
eof = 0                                  /* clear end-of-file flag    */
access_list_count = 0                    /* length of access list cntr  */
space = '                                          '
lastprof = ' '
group_count = 0
say "Profile                       "||,
```

```
    "               Group   Access"
do until eof                            /* loop thru IRRDBU00 unload ds*/
  "execio 100 diskr indd (stem indd."   /* Get some records            */
  if rc = 2 then eof = 1                /* If EOF then set EOF flag    */
  do i = 1 to indd.0                    /* loop thru records read      */
    rtype =  substr(indd.i,1,4)         /* get record type             */
    if rtype = '0100'                   /* Racf group record ?         */
    then call group                     /* yes. the remember it        */
    if rtype = '0404' then              /* data set access list ?       */
      do                                /* yes                         */
        prof  = substr(indd.i,6,44)     /* get data set profile         */
        if lastprof = prof              /* same dsname as last prof ?  */
          then call stack               /* yes. then remember it.      */
          else call print               /* no . print previous prof    */
      end                               /*                             */
  end                                   /*                             */
end                                     /*                             */
return
/*********************************************************************/
/*  List a profile if it has mix of groups and accesses  h */
/*********************************************************************/
print:
  if access_list_count > 1
    then
      do
        access_mix = 0
        do j = 2 to access_list_count
          if  acc.j ¬= acc.1 then access_mix = 1
        end
        if access_mix
          then
            do
              say lastprof grp.1 acc.1
              do j = 2 to access_list_count
                say  space grp.j acc.j
              end
              say space
            end
      end
  lastprof = prof
  access_list_count = 0
  call stack
  return
/*********************************************************************/
/*  Build an array of all groups in access list for a profile        */
/*********************************************************************/
stack:
  this_grp = substr(indd.i,58,8)
  do k = 1 to group_count
      if this_grp = group_name.k
        then
          do
            access_list_count = access_list_count + 1
            grp.access_list_count = this_grp
            acc.access_list_count = substr(indd.i,67,8)
          end
  end
  return
/*********************************************************************/
/*  Build an array of all RACF groups                                */
```

```
/*********************************************************************/
group:
group_count = group_count + 1
group_name.group_count = substr(indd.i,6,8)
return
```

DATASET class profiles

A user or group profile covers only one user or group. Their scope, and thus their impact, is limited to a single entity. Generic profiles, however, have no such limitation. This is the appeal of generic profiles. However, this makes them more complex from a merging perspective.

Assume, for example, that your system data sets are covered by a generic profile SYS1.**. If, during the merge process, you were to add a generic profile for SYS1.LINK*, then **SYS1.LINKLIB** would no longer be covered by the SYS1.** profile. It would now be covered by the SYS1.LINK* profile. The access rules for SYS1.LINKLIB may well have changed. Yet during the DBSYNC compare of the two RACF databases, there would be no indication that this was about to happen. The profiles are not duplicates. Thus, some generic profiles in one database may override profiles in the other, resulting in the risk of either increased or reduced levels of access.

For new RACF groups, added in the earlier step, there is no such problem. The group has been newly added to that RACF database, so there will not be any intersecting DATASET class profiles. DATASET class profiles for these groups can be added without any problems. But for DATASET class profiles for *pre-existing* groups, further checking is required to ensure that these profiles will not cause any unwanted changes to data set access authorities.

The two RACF database unload files can be used to identify this situation and highlight the insertion points of the merged profiles. These profiles need to be checked very carefully to ascertain which data sets will be affected and what that effect will be.

The following topic discusses a particular methodology for adding new DATASET class profiles that required significant programming effort and knowledge of RACF. It is not provided as a recommendation, but as an example of the types of issues that may need to be considered when (a) you are merging two production RACFplexes and (b) it is imperative that you prevent loss of access in either RACFplex at any point during the merge.

There are a variety of ways in which these issues can be handled. The process described here is included for illustrative purposes only, because it was actually used to successfully merge two production RACFplexes for an IBM customer. Because the programs described below contain customer-specific decisions and processing, they are not suitable for general distribution as samples, and are therefore only described at a high level. Note that this particular process requires the use of RACF Enhanced Generic Naming (EGN) for data set profiles on both RACFplexes:

1. A program was written to produce a data set containing one record for each DATASET class profile that existed in either RACFplex. Each profile record contained the following fields:
   - The external name of the DATASET class profile (the name as displayed by or specified on RACF commands).
   - The internal name of the DATASET class profile (the name as stored in the RACF database). The subroutine called RACEX2IN described in 13.4, “Tools and documentation” on page 209 was used to obtain this information.
   - A field indicating which RACFplex the profile was held in.
   - A field indicating whether the profile was generic or discrete.

– A field indicating the first volume serial of the data set for a discrete profile, or blanks if the profile was generic.

– Fields indicating the UACC, WARNING, and ERASE attribute(s) of the profile in each database

> **Note:** The internal-format profile names were used so that profiles not necessarily existing in the same RACF database could be treated as if they all existed in a single database. Once profiles from both databases were sorted in ascending order of internal-format profile name, a program could simulate the results of a RACF search for the best-matching profile in the "merged" data base *before* the actual merge.
>
> This logic was only used for profiles in the DATASET class, since DATASET class profile names cannot contain RACF variables. The rules for RACF variables and resource grouping classes add complexity to RACF processing, prohibiting all but the most determined attempts to simulate RACF's search for the best-matching profile in general resource classes.
>
> If you have access to a "sandbox" (test) RACF database, in which all profiles can be defined prior to the production merge, then there is no need to simulate RACF's search for the best-matching profile. In the project described here, however, a "sandbox RACF" was not used because the risk assessment required access to the ICF catalogs of the production systems, and no existing sandbox system had such access. RACF uses an internal name format that is used to simplify processing of generic and wildcard characters in profiles. The internal format of RACF profile names is partially described in APAR OY31113.
>
> There is one important difference between the internal format of DATASET class profile names and all other general resource profile names—it is related to the fact that, for data sets, an ending single asterisk is more specific than internal double asterisks, while for general resource profiles, the opposite is true. A sample REXX exec containing subroutines for the translation of RACF profile names between internal and external formats is available in the Additional Materials for this book from the Redbooks Web site at:
>
> `http://www.redbooks.ibm.com`

2. The data set produced in step 1 was sorted in ascending order of the internal format profile names.
3. Using this data set as input, you then run a program twice (once on each RACFplex) to evaluate the risk of adding new DATASET class profiles to that RACFplex. The evaluation logic of that program is:
   – Read in the entire sorted file created by step 2, associating each external profile name with its relative position (sequence number) in the input file.
   – Loop through the list of profile names, considering each name individually as follows:
     - If the profile exists only on the current (execution) RACFplex, no further analysis is needed. The impact of adding this profile to the other RACFplex will be determined by the execution of this program on the other RACFplex.
     - If the profile exists in both RACFplexes, compare the UACC, WARNING and ERASE attributes for incompatibilities and report if any are found.
     - If the profile exists only on the other RACFplex, then invoke the Catalog Search Interface, using the external profile name as the search filter, to find all cataloged data sets that could be protected by the new profile. The Catalog Search Interface is described in *z/OS DFSMS:Managing Catalogs,* SC26-7409.

- For each data set returned by the Catalog Search Interface, determine the profile that protects it on the current RACFplex, then compare the sequence number of the profile currently protecting the data set with the sequence number of the profile to be added (which was used as the catalog search filter). If the profile currently protecting the data set has a higher number (because its internal-format name collates higher than the new profile to be added), then this data set will be protected by the new profile in the merged environment.

Note: The Catalog Search Interface filter uses the same masking rules as RACF does for Enhanced Generic Naming (EGN).

- Determining the profile that protects a data set on the current RACFplex can be done by parsing the output from the **`LISTDSD`** command.

Tip: The LISTDSD command always refreshes generic profile information from the RACF database on each call. This method can be quite slow when many data sets are involved.

An alternative that provides better performance was found by writing a command processor to determine the protecting profile using the RACROUTE REQUEST=AUTH macro with the PRIVATE option (which returns the protecting profile name in private storage). Because the DATASET class profiles are processed in collating sequence, generic profiles for any high-level qualifier are only loaded into the private region of the job's address space once. This can save significant amounts of CPU, I/O and time when determining the protecting profile for thousands of data sets.

The source code for this program is called PROPROF and can be located in the Additional Materials for this book on the Redbooks Web site at:

`http://www.redbooks.ibm.com`

- Associate each existing profile that will have data sets “stolen”, with the new profile that “steals” data sets from it. This information is written to a data set and used as input to the profile attribute merge program used in step 4. Also, evaluate the risk of adding the overlapping profile by comparing the UACC, WARNING and ERASE attributes of the “stealing” profile with those of any profile(s) from which it would “steal” protection of existing data sets.
- Assign a risk level to the addition of the new profile. The risk assessment can be determined by factors such as the number of data sets whose protecting profile would change, the compatibility of the attributes of the stealing profile with all profiles it steals from, and any other criteria determined by the installation.

4. The output of the program executed in step 3 was manually reviewed. In addition, an attribute merge program was executed to automatically merge access lists and attributes for overlapping profiles whose risk assessment (as determined automatically in step 3) was below a certain level. In other words, there were many cases where a one-way merge of access list(s) into the stealing profile was deemed safe and was performed automatically.

Important: The issue of overlapping generic profiles is probably the most problematic part of the merge process. Great care is needed to ensure unwanted changes to access levels do not occur.

General resource profiles

General resource class profiles can be generic and so the same considerations apply as those for data set class profiles.

In addition, you should consider the following:

- The inter-relationships between member class and group class profiles are quite complex and must be well understood before performing overlap analysis.
- Profiles containing RACF variables (resolved using profiles in the RACFVARS class) can be especially tricky. It is a good idea to synchronize RACF variables as early as possible during the resolution of general resource profiles.
- Security categories are represented in RACF profiles using a numeric value that indexes members of the CATEGORY profile in the SECDATA class. Use the DBSECLVL REXX exec to post process the RACF database unload utility output prior to using DBSYNC.

13.3.8 Things not handled by DBSYNC

Unfortunately the DBSYNC tool does not cover everything:

1. User passwords are not unloaded by IRRDBU00, and so passwords cannot be synchronized using DBSYNC. See 13.3.9, “Password synchronization” on page 207.
2. RACF maintains Tape Volume Table Of Contents (TVTOC) profiles when the TAPEVOL class and the TAPEDSN option are both active. DBSYNC cannot merge TVTOC profile data, as there are no RACF commands which will update TVTOC profiles. However, inconsistencies are flagged via messages in the SYSTSPRT output from DBSYNC. If TVTOC entries need to be merged, then a program will need to be developed to do this.
3. If the same discrete data set profile exists in both databases (that is, the profile has the same name and volume serial) but the profile is non-VSAM on one, and tape/model/etc. on the other, then DBSYNC will build commands to correct any other discrepancy, but cannot change the profile type. You must handle these situations manually.
4. Many products maintain information in *non-base* segments of RACF user, group and resource profiles. In general, user profiles have the widest variety of possible non-base segments. In some cases, RACF does not provide commands to directly administer the value of non-base segment fields. Additionally, while RACF publications document the names of fields in supported non-base segments, possible field values and their meanings are not always documented by the segment-owning product or component. In some cases, it may be necessary to use facilities provided by the segment-owning product to synchronize, or even meaningfully compare, the corresponding fields of matching profiles. This may or may not require end-user involvement. Another possibility is to update the field directly using an installation-written program.

   The only case of this that we are aware of in an IBM product is the TUPT field of the TSO segment of the RACF user profile. This field holds profile settings that are normally only updated by the TSO **PROFILE** command. While the format of the TUPT is documented in TSO publications, it cannot be updated using normal RACF commands. In order to synchronize the TUPT field, the installation could require each user to review a report of the discrepancies and manually resolve them by logging on to TSO and issuing the appropriate **PROFILE** command. An alternative is to write a program that uses the RACROUTE REQUEST=EXTRACT macro with TYPE=REPLACE to update the TUPT field directly.
5. When comparing profiles, DBSYNC will not consider some differences significant. These differences include items such as profile creation dates, last access dates and times for users, access counts in access lists, and so on. If an installation requires that any of these fields are maintained across the merge, then DBSYNC will need to be modified to

consider these fields. In each unloaded profile, prior to the matching process, DBSYNC replaces these fields with question marks and thus removes their effect from the comparison process.

6. Changing the Group Tree Structure. The addition of a RACF group may alter the RACF group tree structure. If your business processes rely on *group level* privileges, then you should investigate the impact of changes to the group tree structure. For example, if a group of NEW is added with a superior group of OLD, and OLD is an existing group, then users with group privileges in OLD may get authority over group NEW. If this will be a problem, then the RACF Data Security Monitor program (DSMON) can produce a group tree report to allow you to examine the group structure in effect at your installation.

   You should determine what the group structure will look like in the merged environment before actually adding new groups to either database. Groups defined above the insertion point of a new group should be investigated to understand the effect of adding the new group at this point.

13.3.9 Password synchronization

Passwords have three properties that make the database merge difficult:

- They have a high frequency of change.
- Its very obvious when they are incorrect.
- There is no simple way to set them.

Automatic password synchronization requires at least two programs: one to extract the encrypted RACF password field from one RACF database, and another to replace the selected encrypted RACF password field in the target data base.

The RACF Web site contains two utility programs that can be used to unload and reload passwords. The unload data set contains the userid, encrypted password, last password change date and profile creation date fields from the input RACF database. Passwords can be unloaded selectively or en masse. The password unload data set can be processed before being used to update the passwords of the other RACF database, and this allows a degree of flexibility in deciding which passwords to update. The PWDCOPY programs are supplied as load modules only, and so cannot be modified.

The PWDCOPY tool can be found at:

```
http://www.ibm.com/servers/eserver/zseries/zos/racf/pwdcopy.html
```

Use of the PWDCOPY program is recommended to perform the actual updates to the RACF database. User-written programs can be used to manipulate the output from the extract and generate the input to PWDCOPY.

If the PWDCOPY programs do not provide enough functionality, you can develop your own password synchronizing programs. The encrypted RACF password field can be retrieved using the RACROUTE REQUEST=EXTRACT macro with TYPE=EXTRACT, which is described in *z/OS Security Server RACROUTE Macro Reference,* SA22-7692. However, the RACROUTE REQUEST=EXTRACT macro with TYPE=REPLACE cannot be used to update the RACF password using only the encrypted value as input, because RACROUTE assumes

that a replacement value for an encrypted field is provided in clear text and would therefore re-encrypt the value. Instead, use the ICHEINTY ALTER macro in conjunction with an ICHEACTN macro that specifies FIELD=PASSWORD,ENCRYPT=NO to update a RACF password with the encrypted password extracted by the first program.

Note: ICHEINTY and ICHEACTN are documented in *z/OS Security Server RACF Macros and Interfaces,* SA22-7682.

If the user exists in both RACF databases, then the second program will need a method to decide which of the passwords to use.

Some criteria to consider are:

1. If a user has never logged on to a RACFplex, the last access date field of the user record will not be set. In this case, use the password from the other database. User records that were added with the DBSYNC utility will *not* have the last access date and time set.
2. If one of the passwords has been changed more recently than the other, then consider using that password.
3. If the last access date for one password is significantly more recent than the other, then consider using the newer password.
4. If the last access date is older than the password change interval, then consider using the other password.

This method only works if the source password field was encrypted using a technique that the target system supports for decryption.

A suggested method is that if a user has a valid and current password on *both* data sets, then notify the user as to which password will be selected; that is, the password from system X will be used when we cannot determine which password is the best one to use.

The password extraction and update should be done during the outage to re-IPL the necessary systems onto the shared RACF database.

13.3.10 Cutover

Having synchronized the two RACF databases, you can now implement a single RACF database for the entire sysplex. Assuming that you will be moving the incoming system to use the target sysplex RACF database, only the incoming system will need to be IPLed. An outage is not required on the other systems.

The following steps need to be performed:

1. Shut down the incoming system to a minimal state, leaving only those resources active that will be needed to run the final synchronizing tasks.
1. Run a final DBSYNC.
2. Update any remaining discrepancies.
3. Run the password unload utility and extract all passwords.
4. Shut down the incoming system.
5. Update the passwords in the target sysplex RACF database.
6. Update the RACF Data Set Names Table (ICHRDSNT) on the incoming system, specifying the new RACF database.
7. IPL the incoming system.

8. Verify the incoming system is using the correct RACF database using **RVARY LIST**.
9. Monitor syslog and SMF for RACF violations.

13.3.11 Summary

A RACF database merge requires careful analysis of the customization, entities, and resource profiles in both environments. Thorough comparison and reporting of mismatching profile attributes can lead to a methodical approach in which many of the decisions can be made programmatically. By carefully synchronizing the databases using controlled, reversible changes, the merge can be accomplished with minimal risk or disruption, and a high degree of user satisfaction.

Having said that, the process is complex, and mistakes are very visible. For this reason, and especially if you have a large or complex security setup, you may find it advisable to involve someone that has been through a similar process previously. The experiences, and the tools, that they contribute may more than offset the financial cost of their involvement.

13.4 Tools and documentation

In this section we provide information about the tools and documentation that may help you with the merge.

13.4.1 Tools

The following tools are available to perform many of the merge tasks.

The RACF Database Unload utility (IRRDBU00)

The RACF database unload utility enables installations to create a sequential file from a RACF database.

The sequential file can be used in several ways:

- Viewed directly
- Used as input for installation-written programs
- Manipulated with sort/merge utilities
- Uploaded to a database manager, such as DB2, to process complex enquiries and create installation-tailored reports

See *z/OS Security Server RACF Security Administrator's Guide,* SA22-7683 for information on how to use this program.

The RACF Remove ID utility (IRRRID00)

The RACF Remove ID utility (IRRRID00) can be used to remove all references to group IDs and user IDs that no longer exist in a RACF database. In addition, you can optionally specify a replacement ID for those IDs that will be removed.

The remove ID utility:

- Processes the output of the RACF database unload utility (IRRDBU00).
- Uses the DFSORT utility (or an equivalent program) to create lists of IDs from the output of the database unload utility or from user input in a SYSIN file.

- Compares these IDs to the user IDs and group names contained in RACF data fields such as:
  - Standard access list
  - Conditional access list
  - Profile names in certain general resource member classes
  - OWNER fields
  - NOTIFY fields
  - APPLDATA fields of certain general resource profiles
- Generates output consisting of commands that change or remove the references to residual authority:
  - PERMIT commands to delete references on an access list
  - Commands to delete profiles, for profile names that contain the ID value
  - Commands to change references to other values, for references requiring that an ID value be specified

See *z/OS Security Server RACF Security Administrator's Guide,* SA22-7683 for information on how to use this program.

DBSYNC

DBSYNC is a REXX exec that compares two RACF databases, unloaded by IRRDBU00, and creates the RACF commands to make them similar.

To obtain DBSYNC, visit the RACF Web site at:

`http://www.ibm.com/servers/eserver/zseries/zos/racf/dbsync.html`

DBSECLV

DBSECLV is a REXX exec that will read a RACF database unloaded by IRRDBU00, and create a copy of the database with external names added wherever a profile had an internal SECLEVEL or category number.

DBSECLV is intended to be used as a pre-processing step for the DBSYNC exec which must use the external names of SECLEVELs and categories when comparing RACF databases, not the internal SECLEVEL and category numbers.

To obtain DBSECLV, visit the RACF Web site at:

`http://www.ibm.com/servers/eserver/zseries/zos/racf/dbsync.html`

PWDCOPY

The PWDCOPY utility allows you to copy passwords from user IDs in one RACF database to another RACF database. The utility consists of two parts: an export utility (ICHPSOUT) and an import utility (ICHPSIN). The exported file can be post-processed, prior to importing into another RACF database. The exported records contain the userid, encrypted password, last password change date and profile creation date.

To obtain PWDCOPY, visit the RACF Web site at:

`http://www.ibm.com/servers/eserver/zseries/zos/racf/pwdcopy.html`

RACEX2IN

RACEX2IN is a sample REXX exec containing two functions to convert RACF profile names to and from RACF internal name format. If internal-format profiles are sorted in ascending order, then the first (lowest) profile found that matches a resource name is the profile RACF will use to protect that resource. This information can be used to perform RACF analysis on profiles that have not yet been added to a system.

The source code for this exec is located in the Additional Materials for this book on the Redbooks Web site at:

`http://www.redbooks.ibm.com`

PROPROF

PROPROF is a sample TSO/E command processor that displays the name of a profile protecting a resource. This utility performs better than the **`listdsd`** or **`rlist`** RACF commands, because the profile name is retrieved via racroute request=auth, so that RACF database I/O may be avoided. This utility is useful if large numbers of data sets will need to be processed as part of the merge.

The source code for this program is located in the Additional Materials for this book on the Redbooks Web site at:

`http://www.redbooks.ibm.com`

ALTPASS

ALTPASS is a sample TSO/E command processor that allows a user's password to be transferred from one RACF database to another when only the encrypted form of the password is available; that is, when the password itself is not known. ALTPASS also allows a user's revoke count and/or password change date to be updated without changing the user's password. This sample is available as assembler source code and can be used as is, or as a base to develop your own password synchronization process.

The source code for this program is located in the Additional Materials for this book on the Redbooks Web site at:

`http://www.redbooks.ibm.com`

ICETOOL

ICETOOL is a multi-purpose DFSORT utility. ICETOOL uses the capabilities of DFSORT to perform multiple operations on one or more data sets in a single job step.

These operations include the following:

- Creating multiple copies of sorted, edited, or unedited input data sets
- Creating output data sets containing subsets of input data sets based on various criteria for character and numeric field values, or the number of times unique values occur
- Creating output data sets containing different field arrangements of input data sets
- Creating list data sets showing character and numeric fields in a variety of simple, tailored, and sectioned report formats, allowing control of title, date, time, page numbers, headings, lines per page, field formats, and total, maximum, minimum and average values for the columns of numeric data
- Printing messages that give statistical information for selected numeric fields such as minimum, maximum, average, total, count of values, and count of unique values

Tip: SYS1.SAMPLIB member IRRICE contains a lot of useful RACF report generators using ICETOOL.

13.4.2 Documentation

The following manuals contain a wealth of information that can be used during the merge:

- *z/OS Security Server RACF Security Administrator's Guide,* SA22-7683
- *z/OS Security Server RACF Command Language Reference,* SA22-7687
- *z/OS Security Server RACF System Programmer's Guide,* SA22-7681
- *z/OS Security Server RACROUTE Macro Reference,* SA22-7692
- *z/OS Security Server RACF Macros and Interfaces,* SA22-7682

The RACF home page contains useful information and can be found at:

`http://www.ibm.com/servers/eserver/zseries/zos/racf/`

The IBM Redbooks Web site contains a number of RACF-related books and can be found at:

`http://www.redbooks.ibm.com`

14

SMS considerations

This chapter discusses the following aspects of moving a system that is using its own SMS environment into a sysplex where all the existing systems share all or most DASD and have a common SMS environment:

- Is it necessary to merge SMS environments, or is it possible to have more than one SMS environment in a single sysplex?
- A checklist of things to consider should you decide to merge the SMSs.
- A methodology for doing the merge.
- A list of tools and documentation that can assist with this exercise.

14.1 One or more SMSplexes

It *is* possible to have more than one SMSplex in a single sysplex. The definition of an SMSplex is the set of systems that share a single set of SMS CDSs and classification rules (which normally infers all DASD are shared by all systems).

An SMSplex consists of all the systems that share a common SMS ACDS, SCDS, and COMMDS.

If your target environment is a BronzePlex, where you are only sharing the CDSs (and, in particular, not sharing any user volumes), then it is not possible to merge SMS environments.

If your target environment is a GoldPlex, where you would be sharing some system volumes, but no user volumes, you again would probably not merge SMS environments. In order to work effectively, all systems that are in the same SMSplex should really also be in the same RACFplex, HSMplex, RMMplex, and so on. In a GoldPlex environment, you might decide to maintain a single master set of SMS definitions (SMS constructs and routines) that would contain the definitions for both SMSplexes, and do the activate for both SMSplexes from a single SCDS data set; however, you would still have two SMSplexes.

Finally, if your target environment is a PlatinumPlex, where everything, including user volumes, user catalogs, RACF, HSM, RMM, and so on are shared, you would definitely merge SMS environments.

The main benefit of a single merged SMSplex is that you will be able to manage all data in the sysplex from one common central set of policies.

14.2 Considerations for merging SMSplexes

This section contains, in checklist format, a list of things that must be considered should you decide to merge two or more SMSplexes. We say “SMSplexes” rather than SMS control data sets, because there are more things to be considered than simply the contents of the CDSs.

Due to the interrelationship between the following, it is strongly recommended that you do *not* merge the SMS environments unless the following environments are also being merged as part of the process of bringing the incoming system into the target sysplex:

- The HSM environment
- The tape management environment (RMM, for example)
- The security environment (RACF, for example)
- The Master and user catalogs
- Both system and user DASD environments

Table 14-1 Considerations for merging SMSplexes

Consideration	Note	Type	Done
Review Control Data Set sizes and attributes	1	**P**	
Define all systems to the base configuration and review the defaults	2	**P**	
Review all data class, management class, and storage class definitions	3	**P**	
Review all storage group definitions	4	**P**	

Consideration	Note	Type	Done
Review and merge the SMS ACS routines	5	**P**	
Review all tape library definitions	6	**P**	
Review VIO allocations	7	**P**	
Review SMS subsystem definitions	8	**P**	
Review and merge Parmlib member IGDSMSxx	9	**P**	
Review any SMS-related automatic commands	10	**P**	
Review any housekeeping jobs	11	**P**	
Check for SMS exits	12	**P**	
Check if any of the systems are using the allocation exits	13	**P**	
Review the ALLOC00 Parmlib members	14	**P**	
Review the VATLST00 Parmlib members	15	**P**	
Review Esoteric definitions	16	**P**	
Review documentation and recovery procedures	17	**P**	
Check that any required compatibility PTFs are applied if all systems are not using the same DFSMS level		**P**	

The “Type” specified in Table 14-1 on page 214 relates to the sysplex target environment—**B** represents a BronzePlex, **G** represents a GoldPlex, and **P** represents a PlatinumPlex.

Notes in Table 14-1 on page 214 are described below:

1. Review Control Data Set sizes, and attributes.

   Review if control data sets need to be re-sized, and check that the VSAM Shareoptions are specified correctly for shared VSAM data sets. Refer to *DFSMSdfp Storage Administration Reference*, SC26-7331 for more information.

2. Define all systems to the base configuration, and review defaults.

   All systems in the SMSplex must be defined in the base configuration. There is no harm in defining a system in the base configuration before it actually joins the SMSplex. Additionally, it is possible, if you wish, to define all the systems in the sysplex using a single sysgrp definition. If you currently use sysgrp, the incoming system system will automatically be included when it joins the sysplex, so no further action is required.

   You should also review the default values for Management Class, Unit, Device Geometry, and (if running z/OS 1.3 or later) the data set separation profile for both SMSplexes. If the values are different, institute whatever changes are required to bring them into synchronization. Remember that changing the Device Geometry value can cause either space abends, or over-allocation.

3. Review all data class, management class, and storage class definitions.

   Review all the SMS construct definitions, as they will need to be consistent across all systems. This is a critical part of the merge process, and often the most difficult to perform. A simple change to a parameter can have a huge impact on the way data sets and DASD volumes are managed. This, along with reviewing and merging the ACS routines, may be the most time-consuming part of the merge process.

Compare all storage, management, and data classes with duplicate names to see if they really are identical. If they are not identical, you must address this, especially to make sure the wrong data sets don't get deleted.

You also need to check for storage, management, and data classes that are defined with different names, but are in fact identical. You should make a note of any such classes—after the merge is complete, you may wish to change your ACS routines so that only one of the classes gets assigned to new data sets.

The merged SMSplex must contain the superset of storage classes, management classes, and data classes from both SMS environments.

The construct information can be copied from one SCDS to another (assuming both are online to the system you are using) by using the ISMF line operator command COPY. This will bring up a panel that allows you to specify the source and target SCDS. This will copy the constructs, but will not copy the volumes associated with the storage groups. This procedure is documented in *DFSMSdfp Storage Administration Reference*, SC26-7331.

4. Review all storage group definitions.

   Assuming that you are only merging SMSplexes if the target environment is a PlatinumPlex, you need to decide how you want to handle your storage groups. Will you retain the separate storage groups that you have prior to the merge, or will you merge some storage groups, resulting in fewer, but larger, storage groups?

   If you decide to merge storage groups, you need to review the attributes of the storage groups being merged. If the groups being merged have different attributes (for example, one may have AUTOBACKUP set to YES and the other may have it set to NO), you need to ensure the attributes of the merged group meet the requirements of *all* data sets that will reside within that group. Once you have resolved any differences, change the definitions in the target sysplex SMSplex. For simplicity, we do not recommend actually merging the storage groups until after you have brought the incoming system up using the target sysplex ACDS. This means that you must define the incoming system's storage groups in the target sysplex SMSplex.

   In order to allow both SMSplexes to share the one set of SMS constructs while still controlling which volumes each can use, we recommend defining the storage groups so that the incoming system's storage groups have a status of DISNEW for the target sysplex systems, and the target sysplexes' storage groups have a status of DISNEW for the incoming system. This will permit you to have a single SMSplex, but still restrict access to the various storage groups to specific systems. Once you are comfortable that everything is working correctly, you can start changing the storage group's statuses to ENABLE.

   You also need to review which systems you have defined to carry out the various storage management tasks (migration, backup, dump) for each storage group. Many installations limit which systems can do each of these tasks for a given storage group. Given that there are more systems in the sysplex now, and potentially larger storage groups, you should review each storage group definition and decide whether you wish to change the system or system group assigned for each storage group. You obviously need to coordinate this activity with the merge of the HSMplex, to ensure that HSM is enabled for the tasks you wish carried out on the systems you specify in the storage group definition.

5. Review and merge SMS ACS routines.

   This can be the most time-consuming part of the SMS merge. Fortunately, most of the work can be performed prior to the merge day process.

   You need to create a set of SMS ACS routines that will work with both SMS environments. This means creating superset routines containing filter lists, class and group names. You will need to be satisfied that all data sets are being assigned the appropriate Classes and Groups. Consider the use of Naviquest to assist with this task (ISMF option 11).

Avoid deleting the definitions for any data classes, management classes, or storage classes, unless you are sure that they are currently no longer in use. SMS Class names are contained in many places, including catalog entries, HSM CDSs, and on DSS dump tapes. Just because a class is not being assigned to new data sets in the ACS routines does not guarantee that there are no existing data sets with these classes assigned to them. If a management class is deleted, and a data set on Level 0 is assigned to that management class, backup and space management will not process that data set and error messages will be produced saying that the management class could not be found.

If you do eliminate class definitions, consider coding the ACS routines to catch the old names, if data sets are recalled or recovered. Data sets on Level 0 volumes need to have class names defined, but data sets that have been migrated or backed up could have their class names changed by the ACS routines, when they are recalled/recovered to Level 0.

If any systems are using Tape Mount Management, make sure your routines cater for this.

If any systems are using information from the DFP segments, make sure RACF and your ACS routines cater for this.

6. Review all tape library definitions.

   If you have any tape libraries defined to (either) SMSplex, you must ensure that those definitions are carried forward to the merged environment.

7. Review Virtual I/O (VIO) definitions.

   Check to see how VIO is defined in the storage group definitions in both environments. If the definitions are different, make sure you understand why they have different values, and adjust accordingly.

8. Review SMS subsystem definition.

   Review the IEFSSNxx definitions for the SMS subsystem and resolve any differences. The options for the IEFSSN entry is described in the section entitled “Starting the SMS Address Space” in *DFSMSdfp Storage Administration Reference*, SC26-7331.

9. Review and merge Parmlib member IGDSMSxx.

   Review all IGDSMSxx statements for differences, and resolve. The IGDSMSxx statements are described in the section entitled “Initializing SMS Through the IGDSMSxx Member” in *DFSMSdfp Storage Administration Reference*, SC26-7331.

10. Review any automatic or automated commands.

    Review if there is any automation executing SMS-related commands at specific times, or in response to specific messages—for example, SETSMS SAVEACDS(.....). Because all systems will be in the same SMSplex, such commands should probably only be issued on one system in the sysplex.

11. Review any housekeeping jobs.

    Because there will only be one SMSplex, you may be able to eliminate some housekeeping jobs (for example, the jobs that previously ran on the incoming system to back up that system’s SCDS and ACDS).

12. Check for SMS exits.

    You will need to determine what SMS exists are in use, and if they are still actually required. There have been many enhancements to SMS over recent DFSMS releases, meaning that the function provided by your exit may actually be available as a standard part of SMS now. Or maybe the business requirement no longer exists, so the exit can be removed. If you determine that the exit is still required, you need to ensure that the exit

provides the functions required by both the incoming system and the target sysplex systems, and that modifying the exit in preparation for the merge does not have a negative impact on either environment. The SMS exits normal reside in SYS1.LPARLIB and are called:

IGDACSXT	Pre-ACS Installation Exit
IGDACSDC	Automatic Class Selection (ACS) Data class exit
IGDACSSC	Automatic Class Selection (ACS) Storage class exit
IGDACSMC	Automatic Class Selection (ACS) Management class exit

13. Are any of the systems using the allocation exit.

    Determine if other, non-SMS exits (for example, IGGPRE00 or IGGPOST0) are in use. Once again, review whether the exits are still required. If they are, create a superset exit that provides the functions required by both the incoming system and the target sysplex.

14. Review the ALLOCxx Parmlib member.

    Review the ALLOCxx Parmlib member for the SMS-related specifications—for example, SPACE, UNIT and CATLG_ERR. Make sure they are consistent across systems.

15. Review the VATLSTxx Parmlib member.

    This is not strictly an SMS issue, but it is worth reviewing at this time. You should review the VATLSTxx members from the two SMSplexes and create one, merged member that ensures all volumes across both SMSplexes are mounted correctly.

16. Review Esoteric definitions.

    Duplicate esoteric names will probably contain a different set of volumes on each system. These either need to be merged, assuming the data from all systems can coexist and be treated the same (whether SMS or non-SMS), or be renamed and treated differently.

    If they cannot coexist, you should be aware that a rename is potentially serious and affects SMS (if you use Esoteric names in the ACS routines), JCL, and potentially applications, and may not be a simple change.

17. Review documentation and recovery procedures.

    Review documented activation and recovery procedures. Check if CDS names are changing (they will be, for at least the incoming system) or if you need to validate the process for a recovery across multiple systems.

14.3 Methodology for merging SMSplexes

Having done all the preparatory work, there is actually very little involved in moving the incoming system into the target sysplex SMSplex.

You can dynamically move the incoming system into the target sysplex SMSplex by using the SETSMS commands to move the incoming system to the new SCDS, COMMDS, and ACDS data sets. However, as you will doing an IPL anyway to move the incoming system into the target sysplex, we recommend swapping to the new IGDSMSxx member and coming up in the target sysplex SMSplex as part of that IPL.

Once you have successfully tested that the incoming system functions correctly and that data sets get allocated where you would expect them to, you can then start changing the storage groups status to ENABLE. Remember that the user catalogs should be available to all systems at this point, meaning that someone on one of the target sysplex systems could potentially try to recall a data set that had been migrated by HSM on the incoming system, and vice versa. If the incoming system's storage groups are not accessible (ENABLE) to the target sysplex systems, the recall will probably fail.

14.4 Tools and documentation

In this section we provide information about documentation that may help you complete this task. Unfortunately there are no IBM-provided tools to help you in this task; however, the following document contains all the information you should need to ensure a successful merge:

- *DFSMSdfp Storage Administration Reference*, SC26-7331

15

DFSMShsm considerations

This chapter discusses the following aspects of moving a system that is using DFSMShsm into a sysplex where all the existing systems use DFSMShsm and share a set of DFSMShsm CDSs:

- Is it necessary to merge DFSMShsm environments, or is it possible to have more than one DFSMShsm environment in a single sysplex?
- A checklist of things to consider should you decide to merge the DFSMShsm.
- A methodology for doing the merge.
- A list of tools and documentation that can assist with this exercise.

15.1 One or more HSMplexes

It *is* possible to have more than one HSMplex in a single sysplex. The definition of an HSMplex is the set of systems that share a single set of DFSMShsm Control Data Sets (CDSs) and associated HSM-owned DASD and tape volumes. A HSMplex may consist of a number of systems, like the target sysplex, or it may consist of just one system, as in the case of the incoming system.

If your target environment is a BronzePlex, the incoming system will not share any user DASD with the systems in the target sysplex, therefore you would not be merging your HSMplexes.

If your target environment is a GoldPlex, the incoming system would again not be sharing user DASD with the other systems in the target sysplex, so once again you would not be merging your HSMplexes.

Note: If you will be keeping two separate HSMplexes, you need to be aware of a potential for false contention when the HSMs are processing their CDSs. To remove this constraint, DFSMShsm introduced a feature known as “Single GRSplex support”. If you are going to operate more than one HSMplex in a single GRSplex, you will definitely want to exploit this feature. For more information, refer to the section entitled “Single GRSplex Serialization in a Sysplex Environment” in *z/OS DFSMShsm Implementation and Customization Guide,* SC35-0418.

Finally, if your target environment is a PlatinumPlex, the incoming system *will* be sharing the user DASD with the other systems in the target sysplex. In this case, you *would* normally merge the HSMplexes.

While it is possible to do so, we strongly recommend against merging HSMplexes if you are going to retain two distinct pools of DASD. You will need to overcome many problems including coming up with a data set name standard that will avoid duplicate data set names being created, catering for migrated data sets being recalled to the appropriate DASD pool, and unpredictable results when running functions like EXPIREBV.

Note: We assume that there should not be duplicate data set names in the merged environment. Duplicate data set names are a potential integrity exposure, and should always be avoided. In the procedures for merging HSMplexes that we describe later in this chapter, we presume that any cases of duplicate data sets are errors, and that one of the data sets must be deleted or renamed to avoid this situation.

Similarly, we assume if the target environment is a PlatinumPlex, that there should not be any duplicate volsers, either for DASD or tape. If any duplicates are identified during preparation for the HSMplex merge, we assume this is an error situation that will be addressed.

15.2 Considerations for merging HSMplexes

The process of moving from two (or more) HSMplexes to a single HSMplex will take considerable time and planning. Luckily, most of the work can be done prior to the actual merge. In this section we discuss the steps that can be completed in the weeks leading up to the merge. The bulk of this consists of identifying and addressing duplication in the HSM

CDSs. There are also tasks relating to moving to consistent space and availability management practices, setting up new data sets, changing procs and parms, and so on. The actual merge process takes place after all this work has been completed, and is described in 15.3, “Methodology for merging HSMplexes” on page 235.

This section contains, in checklist format, a list of things that must be considered should you decide to merge two or more HSMplexes.

Due to the interrelationship between them, it is strongly recommended that you merge the SMS environment, tape management system (for example, RMM), security environment (for example, RACF), catalog environment, and DASD environments at the same time as merging the HSMplexes.

Table 15-1 Considerations for merging HSMplexes

Consideration	Note	Type	Done
If you will be running with multiple HSMplexes within a single GRSplex, implement “Single GRSplex Support”	1	B, G	
Run an HSM AUDIT on all systems, and perform any required actions, so the systems are clean before starting	2	P	
Identify and eliminate duplicate records within the HSM CDSs	3	P	
Check for duplicate volumes	4	P	
Check for duplicate data sets	5	P	
Check for HSM exits	6	P	
Check that HSM Parmlib and Proclib members have been updated	7	P	
Check that the CDSs and CDS Backup data sets are large enough and have the correct attributes	8	P	
Check if ABARS is being used	9	P	
Check if automation is issuing any HSM commands	10	P	
Review any housekeeping jobs	11	P	
Review security considerations	12	P	
Switch to the new HOSTID in the HSM startup JCL	13	P	
Perform HSM Audit of HSM CDSs	14	P	
Review the chapter entitled “DFSMShsm in a Sysplex Environment” in the *z/OS DFSMShsm Implementation and Customization Guide,* SC35-0418	15	P	
Compatibility PTFs may be required if all systems are not running the same level of DFSMS		P	
If the DFSMShsm software levels are different, and you are sharing a common set of CDSs, it is important to understand which systems can run which functions, and which have particular capabilities—certain functions may only be available on the higher level of HSM		P	
If the HOSTID changes, check that the SMS ACS routines will manage the new Activity Log data sets (which include the HOSTID as part of the name), as expected		P	

Consideration	Note	Type	Done
If using the Common Recall Queue (CRQ) facility, all the members of the HSMplex must have connectivity to the CFs that will contain those structures. To size the CRQ structure, use the CFsizer available at: http://www.ibm.com/servers/eserver/zseries/cfsizer		**P**	

The “Type” specified in Table 15-1 on page 223 relates to the sysplex target environment—**B** represents a BronzePlex, **G** represents a GoldPlex, and **P** represents a PlatinumPlex.

Notes indicated in Table 15-1 on page 223 are described below:

1. If two HSMplexes exist within a sysplex environment, one HSMplex interferes with the other HSMplex whenever DFSMShsm tries to update CDSs in non-RLS mode or when it is performing other functions, such as level 1-to-level 2 migration. This interference occurs because each HSMplex, although having unique resources, uses the same resource names for global serialization.

   To eliminate this constraint, exploit the “Single GRSplex Support” feature in HSM by specifying RNAMEDSN=YES in the startup JCL of every HSM address space, and by specifying the HSMplex name on the PLEXNAME keyword in the ARCCMDxx member. Obviously the two HSMplexes should have different PLEXNAME values.

   For more information on this support, see the section entitled “Single GRSplex Serialization in a Sysplex Environment” in *z/OS DFSMShsm Implementation and Customization Guide,* SC35-0418 and the section entitled “PLEXNAME: Specifying a Name for an HSMplex” in *z/OS DFSMShsm Storage Administration Reference*, SC35-0422.

2. Before you start doing any work with any of the HSM CDSs (on any of the systems), run a HSM AUDIT against all CDSs in both HSMplexes. We recommend running the following audits:

   – AUDIT DATASETCONTROLS(MIGRATION)

   – AUDIT DATASETCONTROLS(BACKUP)

   – AUDIT ABARSCONTROLS

   Running the audit will identify any records that are in error that should be removed or repaired before you start processing the CDSs in preparation for the merge. Running the audit at this time will potentially save you time by avoiding working with records that are irrelevant.

   It would probably also be a good idea at this point to issue a RECYCLE ALL EXECUTE PERCENTVALID(0) HSM command to free up any empty HSM tapes. The fewer entries you have in the HSM CDSs, the easier the merge process should be.

3. Identify and eliminate duplicate records within the HSM CDSs.

Important: There are likely to be quite a number of duplicate records as you review the HSM CDSs. Before you make any changes, you should decide which set of CDSs you want to use as the basis for the merged HSMplex, and which ones you will delete the duplicates from. It may be that you will be forced to make some changes in one HSMplex (for example, relabelling a ML1 volume), and other changes (for example, renaming some data sets) in the other HSMplex, but if you *can* select one HSMplex for all changes, it will make the merge process significantly easier and reduce the opportunity for mistakes to be made.

The merge process that we provide will automatically discard duplicate entries based on the order in which the CDSs were passed to the merge step. If you know that all of the changes you are going to make to remove duplicates will be to HSMPLEX2, then you simply have to include the HSMPLEX2 CDSs as the second data set in the input to the merge. On the other hand, if some of the changes will be made to HSMPLEX1 and some to HSMPLEX2, then you cannot depend on the merge job dropping all the duplicates so you must delete them manually (usually, using the FIXCDS command) prior to the merge.

When merging CDSs, it is important to understand the contents of the data sets. For example, if you are merging the CDSs belonging to two HSMplexes, you cannot do the merge if the same resource is defined in each CDS. You must first analyze the contents of the data sets to ensure that there are no duplicate resources, and remove any that you discover.

The merging of the HSM CDSs will be quite easy provided there are no unexpected duplicate records within them. Some duplicate records can be easily ignored in the merge process, while others need to be reviewed, and the appropriate action taken, before the merge process can proceed.

Job PREMERGE in HSM.SAMPLE.TOOL, which is created by running the job in member ARCTOOLS in SYS1.SAMPLIB, is a job that can assist with determining which records are duplicates. You will need to run this against all CDSs—note that the job only runs against one CDS type at a time, so you will need to run the job three times (once for each set of BCDS, MCDS, and OCDS CDSs).

The output from the PREMERGE job identifies the duplicates by the record number, along with the record key; for example, data set name, tape volser, and so on. There is some documentation within the PREMERGE member that advises on the appropriate action for record types that have duplicates.

An alternate to the PREMERGE job is the “Merge Jobs” for the Migration Control Data Set (MCDS), Backup Control Data Set (BCDS), and Offline Control Data Set (OCDS) (Example 15-1 on page 237, Example 15-2 on page 238, and Example 15-3 on page 239 in 15.4, “Tools and documentation” on page 237).

Just like the PREMERGE job, these jobs can be continually rerun until all duplicates (or at least those that require action) have been resolved. Duplicates will be copied into the data set identified on the DUPS DD statement in Step S002ICET. You may find it more convenient to use the Merge Jobs rather than the PREMERGE job to identify the duplicates, because you will need to set up and run these jobs for the actual merge process anyway.

The record types, the CDS that they reside in, and an indicator of how duplicates should be handled, are displayed in Table 15-2 on page 226, Table 15-3 on page 226, and Table 15-4 on page 226.

Table 15-2 MCDS Records

Type	Description	Note
00	MCD--Migration Control Data Set Data Set Record	**a**
01	MCU--Migration Control Data Set User Record	**b**
02	MC1--Migration Level 1 Free Space Record	**c**
04	MCV--Migration Control Data Set Volume Record	**a**
07	VAC--JES3 Volume Activity Count Record	**b**
10	DSR--Daily Statistics Record	**b**
10	L2CR--Migration Level 2 Control Record	**a**
10	MCR--Management Control Record	**c**
10	MHCR--Multiple-Host Processor Control Record	**c**
10	VSR--Volume Statistics Record	**a**
11	MCA--Migration Control Data Set Alias Entry Record	**a**
12	MCO--Migration Control Data Set VSAM Associations Record	**a**

Table 15-3 BCDS Records

Type	Description	Note
20.00	MCB--Backup Control Data Set Data Set Record	**a**
21.00	DVL--Dump Volume Record	**a**
22.00	DCL--Dump Class Record	**d**
24.00	MCC--Backup Control Data Set Backup Version Record	**a**
26.00	MCM--Backup Control Data Set Move Backup Version Record	**a**
27.00	MCL--BCDS Backup Changed Migrated Data Set Record	**a**
28.00	MCP--Backup Control Data Set Eligible Volume Record	**a**
29.00	DGN--Dump Generation Record	**a**
2A	ABR--Aggregate Backup and Recovery Record	**a**
2C	MCT--Backup Control Data Set Backup Volume Record	**a**
30.00	BCR--Backup Control Record	**c**
30.00	BVR--Backup Cycle Volume Record	**c**
30.00	DCR--Dump Control Record	**c**

Table 15-4 OCDS Records

Type	Description	Note
32.00	TTOC--Tape Table of Contents Record	**d**

The notes listed in these tables have the following meanings:

a. All duplicate records need to be deleted prior to the merge commencing.

b. Duplicate records do not need to be removed prior to the merge; the merge process will automatically discard the duplicates as you run the merge job.

c. Duplicate records do not need to be removed prior to the merge; the duplicates can be safely ignored, *provided you keep the records that correspond to the HSMplex you are merging into*.

d. Duplicate records do not need to be removed prior to the merge, but you do need to check that the duplicates are defined the same on all systems.

There are likely to be several types of HSM CDS records that are contained in both the incoming system and target sysplex CDSs:

– In the BCDS, there are Backup Cycle Volume Records (BVR), Backup Control Records (BCR), Dump Control Records (DCR), and Dumpclass Records (DCL), which are almost certain to be contained in both CDSs.

– In the MCDS, there are Management Control Records (MCR), Migration Level 1 Free Space records (MC1), Migration Level 2 Control Records (L2CR), Multi-host Control Records (MHCR), Migration Control Data Set User Records (MCU), and Daily Statistics Records (DSR).

– In the OCDS, there is only one record type, Tape Table of Contents Records (TTOC), and there won't be any duplicates unless there are tape volumes with the same serial numbers in both HSMplexes.

The following describes each of the CDS record types and discusses how they should be handled.

MCD RECORDS: The MCD record contains migration control information for an individual data set. If there are duplicate MCD records, that is an indicator that there are data sets with duplicate names. All duplicate data sets need to be identified and deleted or renamed prior to the merge—deleting or renaming the data sets will remove the duplicate MCD records.

MCU RECORDS: The MCU record defines the user attributes related to HSM processing. The key of this record is the userid of an authorized storage administrator. Duplicates can be safely ignored, provided the duplicates relate to the same person. If the duplicate owners are really different users, you must decide how to handle that situation.You may have to assign different userids in one of the HSMplexes—in this case, remember to remove the definitions for the userids that have been renamed.

MC1 RECORDS: The MC1 record defines the free space for each available migration level 1 volume in a multiple-host environment. Duplicates will be discarded during the merge job. New entries will be created in the target configuration when the ML1 volume is ADDVOLed.

MCV RECORDS: The MCV record describes a primary or migration volume under DFSMShsm control. Duplicate MCV records indicate that you have duplicate volsers. All duplicate volumes need to be deleted or relabeled prior to the merge, which will result in the duplicate MCV records being deleted.

To relabel a migration volume, you must ADDVOL the volume with the DRAIN attribute, and you must issue the ADDVOL command on every system in the HSMplex that uses that volume. For more information, see the description of the ADDVOL command in *z/OS DFSMShsm Storage Administration Reference*, SC35-0422.

VAC RECORDS: The VAC record exists in a JES3 system and contains counts of the number of times a volume has been returned by DFSMShsm to a JES3 setup request as a candidate for recall of a migrated data set that has not been recalled. Duplicates can be safely ignored, as they will be discarded during the merge job.

DSR RECORDS: The DSR record is a statistical summary of daily activity. Duplicates can be safely ignored, as they will be discarded by the merge job.

L2CR RECORDS: The L2CR record defines the structure of migration level 2 volumes and their associated key ranges. If you are using tape for your ML2 and you have no ML2 DASD volumes, you should not have any of this record type. If there are duplicates, you need to resolve these prior to the merge by emptying and relabelling the DASD ML2 volumes. There are specific considerations for draining ML2 volumes defined with keyranges—see the description of the ADDVOL command in *z/OS DFSMShsm Storage Administration Reference*, SC35-0422 for more information.

MCR RECORDS: The MCR record contains host control information that must be maintained between HSM starts. The key of the record is the characters MCR followed by a single digit—the single digit being the HSM host identification as specified on the HOST keyword in the HSM startup JCL. You do not have to do anything with these records. If the host identification of HSM on the incoming system will change when it moves into the sysplex, the duplicate MCR record will be discarded during the merge job. On the other hand, if the host identification will remain the same (remember that every HSM in the HSMplex must have a different host identification number), then it will be copied correctly to the new merged CDS.

MHCR RECORDS: The MHCR record contains space usage information about the HSM CDSs. You do not have to do anything with these records—duplicate records will automatically be discarded by the merge job. When HSM is started after the merge, using a new set of CDSs, the MHCR record will be updated with information about those data sets.

VSR RECORDS: The VSR record contains information about volume activity for one day for each volume under HSM control. Duplicate VSR records indicate that some volumes under HSM control have duplicate volsers. Obviously you cannot have duplicate volsers after the merge, so all duplicate volumes need to be deleted or relabeled prior to the merge. Removing the duplicate volser ensures that no new duplicate VSR records will be created, and existing ones will be discarded during the merge job.

MCA RECORDS: The MCA record contains information about a migrated data set, including pointers back to the original data set name. You must eliminate any duplicate MCA records before the merge. However, because the key contains the first two qualifiers of the data set and the time and date that it was migrated, it is very unlikely that you will find duplicates. If you do happen to find duplicates, all you have to do is recall the data set and that will cause the MCA record to be deleted.

Note: If you have specified a MIGRATIONCLEANUPDAYS “reconnectdays” value greater than 0, the MCA record will not get deleted until that number of days after the data set is recalled. It might be wise to temporarily set this value to 0 in the days leading up to the merge. Once the merge has been successfully completed, reset reconnectdays back to its original value.

MCO RECORDS: The MCO record contains information about migrated VSAM data sets. The key of the record is the migrated data set name. Just as for the MCA records, there cannot be any duplicate MCO records. However, again like the MCA records, the key contains the date and time and the first two qualifiers of the original data set name, so it is very unlikely that you will actually find any duplicates. If you do, simply recall the data set (with the same reconnectdays consideration as the MCA records).

MCB RECORDS: The MCB record contains control information for an individual data set that has been backed up, and identifies backup versions. The key is the original data set name, so duplicate records are an indicator that there are duplicate data set names. You cannot have duplicate data sets in the merged HSMplex, so all duplicates data sets need to be deleted or renamed (and re-backed up) prior to the merge. Once you delete or rename the data sets, the duplicate MCB records should disappear.

DVL RECORDS: The DVL record contains information about HSM dump volumes. The key is the dump volume volser, so duplicate DVL records indicate that there are duplicate dump volsers. You must eliminate any duplicate volsers before the merge (and before you merge tape management systems), so all duplicate volumes need to be deleted or renamed prior to the merge. Once the volumes are DELVOLed, the duplicate DVL records will be deleted.

DCL RECORDS: The DCL record describes the attributes of a HSM dump class. Duplicates can be safely ignored (the duplicates will be discarded during the merge job), provided the related dump classes are defined the same on each system. If the duplicates are different, then you should review and consolidate the dump class definitions in ARCCMDxx.

MCC RECORDS: The MCC record describes a backup version of a data set. The key contains the backup data set name, and, like the MCA records, it contains the first two qualifiers of the original data set and the date and time that it was backed up. It is theoretically possible, although unlikely, that you will have duplicate MCC records. If you do, you must delete the related backup data set using the BDELETE HSM command. Before you do this, however, you need to be sure that that backup is no longer required. After you issue the BDELETE commands, the duplicate MCC records should no longer exist.

MCM RECORDS: The MCM record describes a data set that has been backed up by the BACKDS or HBACKDS command, and is currently residing on a level 1 migration volume. It is rare that any will exist, and it is very unlikely that any duplicates will also exist. You should issue a “FREEVOL ML1BACKUPVERSIONS” command prior shutdown to move all these backup data sets from ML1 to tape. When this command completes, all the MCM records should have been deleted.

MCL RECORDS: The MCL record describes a changed data set that has migrated from a primary volume, and needs to be backed up. It is rare that any will exist, and it is very unlikely that any duplicates will also exist. The key is the data set name, so duplicate keys are an indicator of duplicate data set names. If you eliminate all duplicate data set names, there should be no duplicate MCL records.

MCP RECORDS: The MCP record contains information about volumes that have been backed up or dumped by DFSMShsm. The key is the volume serial number, so if you have duplicate MCP records, it is an indicator that you have, or had, duplicate volsers. While you should resolve the duplicate volser situation, that will not remove the duplicate MCP records. The records will only be removed when all backups and dumps have been deleted. Because the backups or dumps may contain information that is needed, you must decide how you want to handle each one. We do not recommend proceeding with the merge if there are still duplicate MCP records, as the merge process could discard information about backups that you still need.

DGN RECORDS: The DGN record contains information about the dump generation of a given volume when this volume has been processed by the full-volume dump function. The key is the DASD volser followed by the date and time the dump was created. If you have duplicate records, it is an indication that you have duplicate DASD volsers. While you should resolve this before the merge, that action on its own will not remove the duplicate DGN records. The DGN records will only be deleted when the related dump is deleted. As

for the MCP records, care must be taken when deciding to delete dump volumes in case they contain information that is still required. However, you cannot eliminate the DGN records until you delete the related dumps. We do not recommend proceeding with the merge until all the duplicate DGN records have been addressed.

ABR RECORDS: The ABR record contains information related to a specific version and copy created during an aggregate backup or processed during an aggregate recovery. It is unlikely there will be any duplicate ABR records. However, if there are:

- Duplicate AGGREGATE names should be identified as part of the preparation for the merge of the DFSMS SCDS constructs. If duplicates are found, they need to be resolved at this point.
- If no duplicate aggregate names are found in the SCDSs, but there *are* duplicates in the BCDSs, then it is likely that one site has some dormant ABR records that can probably be omitted from the merge—however, you must verify this before proceeding. If the aggregates are required, and the duplicate records are not addressed, you will end up losing versions of the aggregate you want to keep.
- For any duplicates that are identified, you need to determine:
  - Are the applications included in the aggregate intended to be merged?
  - If the applications *are* being merged, you need to decide which ABR records to keep.
  - If applications are not being merged, you have to decide which application will keep the aggregate name and decide on a new aggregate name needed for other application.
  - In the latter case, you must set up definitions and establish an ABR history for the new aggregate (by using ABACKUP). Once you have done this, the duplicate ABR records can probably be dropped prior to or during the merge.

MCT RECORDS: The MCT record describes a volume used for containing backup versions. The key of these records is the volser of the backup volume, so duplicate MCT records indicate duplicate backup tape volsers. The duplicate volsers must be addressed, probably by doing a RECYCLE followed by a DELVOL of the tapes in question. However, to get the best value from the RECYCLE, you should do an EXPIREBV prior to running the RECYCLE—this will ensure that all expired data is deleted before you clean up the tapes. Once the DELVOL has been issued, there should no longer be any duplicate MCT records.

BCR RECORDS: The BCR record contains status information about backup processing. Each HSM host that is allowed to run backup (that is, SETSYS BACKUP) has a defined host identification (specified in the HOST keyword in the startup JCL) and a unique BCR record. The duplicate BCR records will automatically be discarded during the merge job, so you don't have to do anything with these records.

BVR RECORDS:

Important: Unlike the other CDS records, where duplicate records should be addressed well in advance of the merge, the process for fixing up the BVR records *must* be carried out at the time the CDSs are merged. We include the information here because it logically fits in with the discussion in this section; however, the actions listed here should not be carried out until the time of the merge.

The BVR record describes both tape and DASD backup volumes that are: (1) assigned for use on a particular day of the backup cycle; (2) to be used for spill processing; and (3) unassigned.

Figure 15-1 shows how the BVR records are represented in the BCDS.

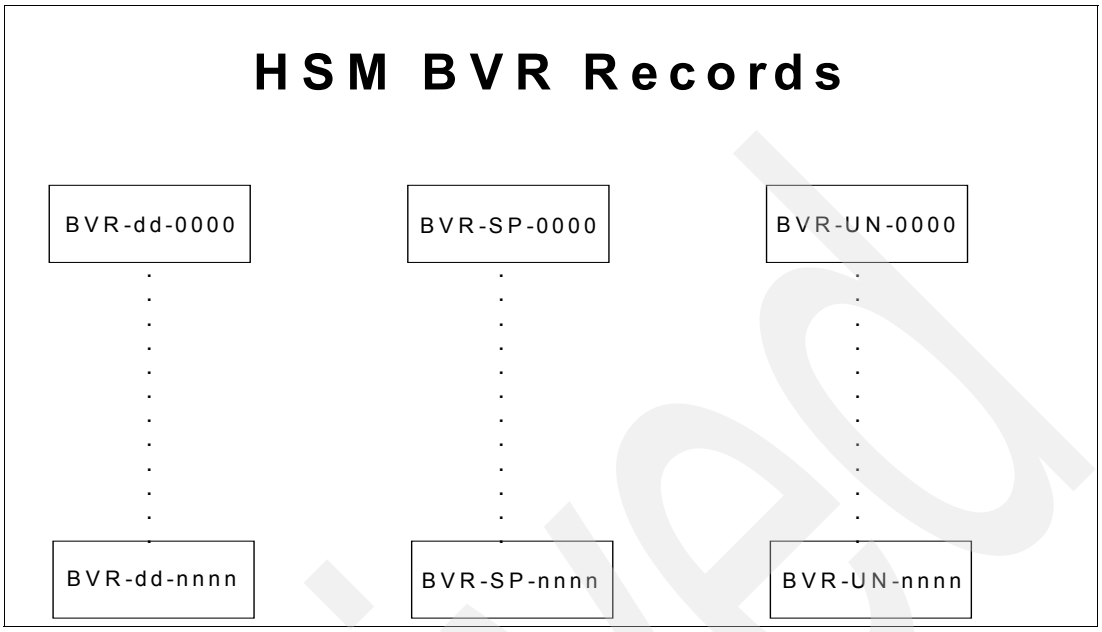


Figure 15-1 Example of Backup Cycle Volume (BVR) Record Structures

Each of the BVRs shown in Figure 15-1 constitutes a day in the daily backup cycle (where “dd” is 01 through the number of days defined by the DEFINE BACKUPCYCLE), or “SP” for spill volumes, or “UN” for unassigned volumes. Each BVR record holds up to 164 entries. So, for example, if you had 180 tapes assigned to DAY 1, you would have a record with a key of BVR-01-0000 containing 164 entries, and another one with a key of BVR-01-0001 containing 16 entries.

Because of this naming convention for the BVR records, you will almost definitely have duplicate BVR records across the two HSMplexes. Fortunately, APARs OW37110 and OW35561 have made it significantly easier to merge CDSs containing these records—so much easier, in fact, that if you do not have those APARs applied, we recommend waiting until these APARs are applied to both HSMplexes before you start the merge.

Once these APARs are applied, the process you use is as follows:

a. DELVOL ... BACKUP(MARKFULL) all partial backup tapes on the incoming system prior to the merge. If you are already using MARKFULL, this can be ignored.

b. Stop all HSM address spaces.

c. Back up the CDSs of all systems.

d. Merge the BCDSs. Duplicate BVR records do not need to be removed prior to the merge; the duplicates can be safely ignored, provided you keep the records that correspond to the system you are merging into.

e. Start one of the HSMs using the new merged databases.

f. Issue a FIXCDS R BVR REFESH(ON) HSM command from that HSM. HSM will automatically rebuild the BVR records to include information for all partially filled backup volumes, regardless of which BCDS they originated in.

g. Issue the BACKVOL CDS command to get another backup of the CDSs.

Note: If you don't have APARs OW37110 and OW35561 applied to your systems, and you can't apply them for some reason, you should use the following process. The objective of this step is to move volumes from one BVR to another so that each BVR structure is unique to each configuration.

The way to move volumes from one BVR to another is to use the ADDVOL and DELVOL commands. DELVOL UNASSIGN moves a volume from a daily backup or spill BVR into the unassigned BVR. ADDVOL can move a volume from the unassigned BVR into any of the daily backup or spill BVRs.

Thus, if both of the HSMplexes are running with seven-day backup cycles, the storage administrator should DELVOL UNASSIGN all the daily backup volumes from one of the HSMplexes and ADDVOL them to BVR01. The spill volumes would be handled similarly, moving them from BVRSP to BVR02. The volumes that were originally unassigned can be ADDVOLed to BVR03. The storage administrator would repeat the process in the other HSMplex, using BVR04, BVR05, and BVR06.

When the backup volumes are moved, the residual empty BVR records on each of the configurations should be deleted with the FIXCDS command.

DCR RECORDS: The DCR record contains control information for dump processing. Each HSM host that is allowed to run dump has a defined host identification and a unique DCR record. The duplicate DCR records will automatically be deleted by the merge job, so you don't have to do anything with these records.

4. Are there duplicate DASD or tape volsers?

   There may be DASD and/or tape volumes that exist in both HSMplexes. If so, it is likely that you will have discovered them as you reviewed the duplicate CDS records. However, it is possible that there are duplicate volumes that HSM was not aware of at the time you reviewed the CDSs.

   To check for duplicate DASD volsers, get a list of all the DASD from both HSMplexes and use a tool such as SUPERC to check for any matches.

   If you find a duplicate DASD volume, you can use something like DFDSS COPY to move Level 0-resident data sets to another volume with a unique volser. If the duplicate volume is an ML1 volume, you should use FREEVOL to move ML1 and ML2 DASD-resident data sets to another volume with a unique volser.

   To check for duplicate tape volumes, check the tape management systems from both HSMplexes. There should be a tape management system merge project going on in tandem with the HSMplex merge, and identifying duplicate volsers will be one of the first tasks of that project. It is unlikely that you will find HSM-owned duplicate tape volumes that were not already discovered during the CDS cleanup process; if you do find any, go back and review the steps for the related record type. Duplicate tapes that are not owned by HSM will be addressed by the tape management system project. If you do find a duplicate HSM tape volume, RECYCLE that tape to move the contents to another volume with a unique volser.

   After you have freed up the duplicate volumes, issue a DELVOL PURGE command to remove the volumes from HSM. If the duplicate volume is a Level 0 volume, delete the HSM VTOCCOPY data sets (there should be two of these for each Level 0 volume with the AUTOBACKUP attribute, then delete the MCP record for these volumes.

5. Are there duplicate data set names?

   It is important to understand that you need to resolve duplicate data sets not just between HSMplexes, but also between the overall catalog environments of all systems. If you do not resolve all duplicate data set names, there are many problems that can occur, in addition to the HSM-related ones.

   It is possible that some data set names may exist in both configurations. Some of these are to be expected, like SYS1.LINKLIB. These will not cause you a problem as you will be moving to a single shared sysres as part of the merge. However, if there are user data sets with duplicate names, then you will have to either delete or rename one or the other. As with duplicate volumes, many of the duplicates should have been discovered as you reviewed the HSM CDSs.

   However not all data sets are necessarily managed by HSM, so there may be duplicates in addition to any you have already found. To get a list of those, a good place to start is by reviewing the aliases in the two Master Catalogs (hopefully all data sets are cataloged in the standard search order!). If there are duplicate aliases, then there is the possibility of duplicate data sets. You could also run DCOLLECT against all non-system volumes in both HSMplexes, followed by ICETOOL to identify any duplicates (similar to the way we identified the duplicate CDS records). Any duplicates you discover are going to have to be addressed (not forgetting the effect on things like RACF, JCL, backups, and so on).

6. Check HSM exits

   You will need to determine which HSM exits are in use, and to verify that the exits are exactly the same on all systems. Take this opportunity to review if all the active exits are actually still needed. Issue the QUERY SETSYS HSM command on all systems in both HSMplexes to identify which HSM exits are in use.

7. Have HSM Parmlib and Proclib members been updated?

   To allow for a simplified backout process, it is recommended that you do not change the current HSM parmlib members or HSM startup JCLs, but rather create new merged versions. On the actual merge day, you can simply rename the old ones, and rename the new ones to perform the merge.

   The following considerations apply to the ARCCMDxx Parmlib member:

   – Decide whether you are going to use separate ARCCMDxx members, or one combined version of ARCCMDxx for all HSM subsystems, utilizing the ONLYIF command if there are parameters that only apply to one subsystem.
   – Review all ARCCMDxx statements for differences, and resolve. You should issue a QUERY SETSYS command on each system; this will make this task considerably easier, because when you compare the output, all messages will be in the same order.
   – Review all Auto Function start times and end times, as these may now require more time to complete, or there may be more tasks to complete in the same period.
   – Check that all UNITNAMES for backup, migration, and so on exist and are accessible from all systems.
   – You may wish to use new names for the CDS backups (specified on the CDSVERSIONBACKUP) keyword. This will ensure that the pre-merge backups are available should you need to backout.
   – Merge all the ADDVOL statements for the non-SMS and HSM ML1 volumes.
   – Review all DUMPCLASS definitions and merge them as appropriate (any duplicate DUMPCLASS definitions should have been addressed when you were reviewing the CDSs).
   – Review the definitions of the HSM authorized users (AUTH statements) and merge.
   – To enable a controlled startup after the merge, you may wish to add statements to HOLD all functions, and start up in EMERGENCY mode.

– If you change the MIGRATEPREFIX and BACKUPPREFIXs as part of the merge, after the merge you will still be able to access the migrated and backup data sets that were created before the merge; however, newly-created migrated and backup versions will use the new HLQs.

Note: Recalls from the SDSP data set will not work because for SDSP the system doing the recall reconstructs the SDSP file name using the current MIGRATEPREFIX value. HSM does not record the actual name of the SDSP anywhere.

In addition, recalls or recovers from tape may be an issue if you are running an OEM tape package that compares the full 44-byte name on the allocation request to what it has recorded in its records. Some tape packages specifically do this, although they provide an option to turn it off. This is not an issue with RMM.

You should also review the HSM startup JCL:

– Decide whether you are going to use separate HSM Procs for each system, or one version that utilizes symbols defined in IEASYMxx.
– Update the HOST parameters as appropriate. You will more than likely have to change the host identification of the incoming system. Review if this parameter should be specified via a symbol defined in IEASYMxx.
– Make sure each HSM system has unique data sets for ARCPDOX/Y & ARCLOGX/Y. We suggest that you include the HSM address space name and the system name (&SYSNAME) in the data set name.
– Review the CDSR & CDSQ values—they may need to be updated to support sharing of the CDSs. Refer to the chapter entitled “DFSMShsm in a Multiple-Image Environment” in *z/OS DFSMShsm Implementation and Customization Guide,* SC35-0418. This may require updates to the GRS RNLs which you will need to coordinate to take effect at the same time as you merge the CDSs.

8. Are the CDSs and the CDS backup data sets large enough for the merged configuration and do they have the correct attributes?

   To allow for a simplified backout process, we recommend that you create new versions of the CDSs and Journal. On the actual merge day, you simply rename the HSM procs so that you use the new procs that refer to the new names.

   You should calculate the required sizes for the new, combined CDSs based on information from the CDSs that will be merged, and allocate the CDSs. At the same time, calculate the size for the new Journal data set, and allocate the new Journal data set.

   Resize the CDS backup data sets to accommodate the new CDS and Journal sizes. Remember that HSM will reuse and rename existing DASD-resident backup data sets when it backs up the CDSs or Journal, so if you want to switch to larger backup data sets, you will have to allocate the new backup data sets in advance. You must define as many backup data sets for each CDS as are specified on the BACKUPCOPIES statement in ARCCMDxx.

   Review the HSM-provided job to allocate the CDSs (HSM.SAMPLE.CNTL(STARTER) which is created by member ARCSTRST in SYS1.SAMPLIB), to check that you are defining the new CDSs with the correct record size, bufferspace, CIsize, Shareoptions, and so on.

9. Is ABARS used?

   If both HSMplexes use ABARS, check for duplicate Aggregate names. If duplicates exist, only one system will be able to keep the original name. It may be a complex task to remove the duplicates if it is determined that you must maintain past data from both aggregate groups.

10. Is automation issuing any HSM commands?

    Check if there are any HSM commands being executed by automation, either at specific times, or in response to specific HSM messages. If so, you may wish to change automation to run these from only one of the systems in the HSMplex.

11. Review any housekeeping jobs

    Review any housekeeping jobs, and confirm that data set names match those that will be used after the merge. It may now be desirable to run these from only one of the systems in the HSMplex.

12. Review HSM security definitions

    Chapter 9, “Authorizing and Protecting DFSMShsm Resources” in *z/OS DFSMShsm Implementation and Customization Guide,* SC35-0418, discusses security for DFSMShsm. If the systems to be merged are also merging the RACF environment at the same time, and you have set up the Parmlib definitions the same on all systems, there should be no additional security concerns. Make sure all ARCCMDxx SETSYS definitions for TAPESECURITY are defined the same way.

13. Switch to new HOSTIDs

    If possible, switch all HSMplex systems to use the HOSTIDs that they will use in the merged sysplex prior to the merge.

    If you are able to do this, you can delete the old control records that have the old HOSTID (that is, the BCR, DCR, and MCR records), and execute a LIST HOST(old hostid) RESET command to remove references to the old HOSTID.

14. Audit HSM CDSs

    Just before you shutdown to do the merge, do another HSM AUDIT of all CDSs (in both HSMplexes), and perform all the appropriate cleanup actions. This will ensure all records are correct and accurate before starting the merge.

15. Review Chapter 12 entitled “DFSMShsm in a Sysplex Environment” in *z/OS DFSMShsm Implementation and Customization Guide,* SC35-0418 for an understanding of the additional HSM functions available in a sysplex environment.

    You should also review the Secondary Host Promotion feature, which provides the ability for one of the members of the HSMplex to take over as the Primary HSM should the existing Primary HSM fail or be stopped.

15.3 Methodology for merging HSMplexes

In this section we discuss the approaches that can be taken to merging HSMplexes, and give some advice about the approach that provides the smallest risk or is the easiest to implement.

There are a number of different techniques for merging HSMplexes. In this Redbook, we present one that is low risk and relatively easy to do. An alternate approach to the one discussed here is to use ABARS. The advantages of using ABARs are:

1. You can change the HLQ of any L0 data set and migrated non-VSAM data set (without recalling it).
2. You can complete the merge without requiring an HSM outage on the target HSMplex.

The disadvantages of using ABARS are:

1. Backup data is not copied (only L0 ML1 & ML2).
2. You need to read and rewrite all ML2 volumes.

The method we provide in this book presumes that a planned HSM outage is possible.

15.3.1 Merging the CDS

Follow the steps outlined below to perform the merging of CDSs. This assumes that all the actions identified in 15.2, “Considerations for merging HSMplexes” on page 222 have been completed. If any potential problems or duplicate records are identified, you must make the recommended changes before continuing**.**

1. Select a window when no HSM automatic functions are scheduled. If possible, try to pick a time when there will be no HSM requests from TSO user or batch jobs either.
2. Before you shut down, carry out any steps necessary to prepare the BVR records for the merge. Refer to “BVR RECORDS” on page 230.
3. Issue a RECYCLE ALL EXECUTE PERCENTVALID(0) command to free up any empty HSM-owned tapes.
4. After the RECYCLE completes, issue a SETSYS EMERGENCY command on all HSMs in both HSMplexes. This will stop any new HSM requests from starting, but allow any current processing to finish.
5. Issue a BACKVOL CDS command one on system in each HSMplex if you intend to use this backup as an additional form of backout. Being in EMERGENCY mode will ensure that no HSM processing takes place after this command completes.
6. Shut down all the HSM address spaces that are to be merged, using the HSM STOP command, and allow them to stop normally.
7. Run the jobs “MCDS Merge Job”, “BCDS Merge Job”, and “OCDS Merge Job” located in 15.4, “Tools and documentation” on page 237.

   Review the output from each job, including the records contained in the data set specified on the Step2 DUPS DD statement. The records in that data set are duplicate records that will not be merged. Note that when duplicates are discovered in the sorted input file (the one with the DDNAME of RECS), the first occurrence will be retained, so the order of DD statements in the PRIM DD in step S002ICET should place the system that contains the records you wish to retain first.

   While it is normal to have some records in the DUPS data set, make sure that the duplicate records are the ones you are expecting (based on the information in 15.2, “Considerations for merging HSMplexes” on page 222). If you find any unexpected duplicates, you will need to restart one of the related HSMs, use the instructions provided in 15.2, “Considerations for merging HSMplexes” on page 222 to clean up the duplicates, and start from Step 1 above again.
8. Rename the HSM ARCCMDxx member to pick up the one for the merged HSMplex. Also, rename the HSM startup JCL member now to pick up the one that refers to the new CDSs and Journal.
9. Start up the new merged HSM, initially on just one system. Review the system log for syntax errors in the HSM parms PARSE errors. If errors exist, fix them in Parmlib, and restart HSM until it is clean.
10. Before proceeding, complete the procedure for merging the BVR records. Refer to “BVR Records’ on page 230 for the details.
11. Perform testing and check that the merged information is good by using DFSMShsm commands.
12. After successful testing, perform a BACKVOL CDS.
13. Update the HSM ARCCMDxx member to remove the HOLD and SETSYS EMERGENCY statements.

14. Consider running an AUDIT command with the NOFIX option, to display any errors.

15.3.2 Backout

If backout is required at any stage, because you created new versions of the CDSs, Journal, ARCCMDxx Parmlib member and startup JCL, you can simply stop all the HSMs, rename the Parmlib and Proclib members back to their original names, and restart all the HSMs.

15.4 Tools and documentation

In this section we provide information about tools and documentation that may help you complete this task.

The following manuals will be required during the merge project:

- *z/OS DFSMShsm Diagnosis Reference,* LY35-0115 contains the layouts of the HSM CDS records.
- *z/OS DFSMShsm Implementation and Customization Guide,* SC35-0418 contains information about setting up HSM, and specifically information about considerations for a sysplex environment.
- *z/OS DFSMShsm Storage Administration Reference*, SC35-0422 contains information about the HSM console commands, and the HSM statements that are specified in the ARCCMDxx member.

Example 15-1, Example 15-2 on page 238, and Example 15-3 on page 239 contain jobs to merge the CDSs. The first step of each job unloads a given pair of CDSs (BCDS, MCDS, or OCDS) into a flat file. The second step sorts the flat file, then selects records from the sorted file into another flat file. If any duplicates are found, the one from REPRO1 will be written to the flat file, and the one from REPRO2 will be discarded and written to the DUPS file. Finally, the new, merged, CDS is allocated and loaded with the records from the flat file that was created by the sort.

Important: It is *vital* to specify the "right" CDS on the REPRO1 statement in each job. If there are any duplicate records, the one from REPRO1 will be selected, and the ones from REPRO2 will be discarded.

In addition, it is vital that you are consistent about which CDS you specify on REPRO1 in the three merge jobs. Either specify the incoming system CDS on REPRO1 in all jobs, or else specify the target sysplex CDS on REPRO1 in all jobs. If you specify the target sysplex CDS on some REPRO1s, and the incoming system CDSs on others, the results will be unpredictable.

Example 15-1 MCDS Merge job

```
//SO01AMS  EXEC PGM=IDCAMS,REGION=0M
//SYSPRINT  DD  SYSOUT=*
//MCDS1     DD  DISP=SHR,DSN=Input.MCDS1
//MCDS2     DD  DISP=SHR,DSN=Input.MCDS2
//REPRO1    DD  DSN=&&REPRO1,DISP=(,PASS),
//           UNIT=SYSALLDA,SPACE=(CYL,(200,20),RLSE),
//           LRECL=2040,RECFM=VB,DSORG=PS
//REPRO2    DD  DSN=&&REPRO2,DISP=(,PASS),
//           UNIT=SYSALLDA,SPACE=(CYL,(200,20),RLSE),
//           LRECL=2040,RECFM=VB,DSORG=PS
//SYSIN     DD  *
```

```
   REPRO IFILE(MCDS1) OFILE(REPRO1)
   REPRO IFILE(MCDS2) OFILE(REPRO2)
/*
//*--------------------------------------------------------------------
//SOO2ICET EXEC  PGM=ICETOOL
//TOOLMSG  DD SYSOUT=*
//DFSMSG   DD SYSOUT=*
//DFSPARM  DD *
VLSHRT
/*
//DUPS     DD SYSOUT=*
//PRIM     DD DISP=SHR,DSN=&&REPRO1
//         DD DISP=SHR,DSN=&&REPRO2
//RECS     DD DSN=&&RECS,REFDD=*.PRIM,SPACE=(CYL,(1,10),RLSE),
//          UNIT=SYSALLDA
//MERGED   DD DSN=&&REPROM,DISP=(,PASS),
//          REFDD=*.PRIM,SPACE=(CYL,(1,10),RLSE),
//          UNIT=SYSALLDA
//TOOLIN   DD *
  SORT FROM(PRIM) TO(RECS) USING(RECS)
  SELECT FROM(RECS) TO(MERGED) ON(5,44,CH) FIRST -
   DISCARD(DUPS)
//RECSCNTL DD *
  OPTION EQUALS
  SORT FIELDS=(5,44,CH,A)
/*
//*--------------------------------------------------------------------
//SOO3AMS  EXEC PGM=IDCAMS
//SYSPRINT  DD  SYSOUT=*
//MERGED    DD  DISP=SHR,DSN=&&REPROM
//SYSIN     DD  *
   DEFINE CLUSTER (NAME(New.Merged.MCDS) VOLUMES(vvvvvv) -
        CYLINDERS(nnn 0) -
        RECORDSIZE(435 2040) FREESPACE(0 0) -
        INDEXED KEYS(44 0) SHAREOPTIONS(3 3) -
        SPEED BUFFERSPACE(530432) -
        UNIQUE NOWRITECHECK) -
        DATA(NAME(New.Merged.MCDS.DATA) -
        CONTROLINTERVALSIZE(12288)) -
        INDEX(NAME(New.Merged.MCDS.INDEX) -
        CONTROLINTERVALSIZE(2048))
   REPRO IFILE(MERGED) ODS(New.Merged.MCDS)
/*
```

Example 15-2 BCDS Merge job

```
//SOO1AMS  EXEC PGM=IDCAMS,REGION=OM
//SYSPRINT  DD  SYSOUT=*
//BCDS1     DD  DISP=SHR,DSN=Input.BCDS1
//BCDS2     DD  DISP=SHR,DSN=Input.BCDS2
//REPRO1    DD  DSN=&&REPRO1,DISP=(,PASS),
//           UNIT=SYSALLDA,SPACE=(CYL,(200,20),RLSE),
//           LRECL=6544,RECFM=VB,DSORG=PS
//REPRO2    DD  DSN=&&REPRO2,DISP=(,PASS),
//           UNIT=SYSALLDA,SPACE=(CYL,(200,20),RLSE),
//           LRECL=6544,RECFM=VB,DSORG=PS
//SYSIN     DD  *
```

```
   REPRO IFILE(BCDS1) OFILE(REPRO1)
   REPRO IFILE(BCDS2) OFILE(REPRO2)
/*
//*--------------------------------------------------------------------
//SOO2ICET EXEC  PGM=ICETOOL
//TOOLMSG  DD SYSOUT=*
//DFSMSG   DD SYSOUT=*
//DFSPARM  DD *
VLSHRT
/*
//DUPS     DD SYSOUT=*
//PRIM     DD DISP=SHR,DSN=&&REPRO1
//         DD DISP=SHR,DSN=&&REPRO2
//RECS     DD DSN=&&RECS,REFDD=*.PRIM,SPACE=(CYL,(1,10),RLSE),
//          UNIT=SYSALLDA
//MERGED   DD DSN=&&REPROM,DISP=(,PASS),
//          REFDD=*.PRIM,SPACE=(CYL,(1,10),RLSE),
//          UNIT=SYSALLDA
//TOOLIN   DD *
  SORT FROM(PRIM) TO(RECS) USING(RECS)
  SELECT FROM(RECS) TO(MERGED) ON(5,44,CH) FIRST -
   DISCARD(DUPS)
//RECSCNTL DD *
  OPTION EQUALS
  SORT FIELDS=(5,44,CH,A)
/*
//*--------------------------------------------------------------------
//SOO3AMS  EXEC PGM=IDCAMS
//SYSPRINT  DD  SYSOUT=*
//MERGED    DD  DISP=SHR,DSN=&&REPROM
//SYSIN     DD  *
   DEFINE CLUSTER (NAME(New.Merged.BCDS) VOLUMES(vvvvvv) -
        CYLINDERS(500 0) -
        RECORDSIZE(6544 6544) FREESPACE(0 0) -
        INDEXED KEYS(44 0) SHAREOPTIONS(3 3) -
        SPEED BUFFERSPACE(530432) -
        UNIQUE NOWRITECHECK) -
        DATA(NAME(New.Merged.BCDS.DATA) -
        CONTROLINTERVALSIZE(12288)) -
        INDEX(NAME(New.Merged.BCDS.INDEX) -
        CONTROLINTERVALSIZE(2048))
   REPRO IFILE(MERGED) ODS(New.Merged.BCDS)
/*
```

Example 15-3 OCDS Merge job

```
//SOO1AMS  EXEC PGM=IDCAMS,REGION=0M
//SYSPRINT  DD  SYSOUT=*
//OCDS1     DD  DISP=SHR,DSN=Input.OCDS1
//OCDS2     DD  DISP=SHR,DSN=Input.OCDS2
//REPRO1    DD  DSN=&&REPRO1,DISP=(,PASS),
//           UNIT=SYSALLDA,SPACE=(CYL,(200,20),RLSE),
//           LRECL=2040,RECFM=VB,DSORG=PS
//REPRO2    DD  DSN=&&REPRO2,DISP=(,PASS),
//           UNIT=SYSALLDA,SPACE=(CYL,(200,20),RLSE),
//           LRECL=2040,RECFM=VB,DSORG=PS
//SYSIN     DD  *
   REPRO IFILE(OCDS1) OFILE(REPRO1)
```

```
   REPRO IFILE(OCDS2) OFILE(REPRO2)
/*
//*-------------------------------------------------------------------
//SOO2ICET EXEC  PGM=ICETOOL
//TOOLMSG  DD SYSOUT=*
//DFSMSG   DD SYSOUT=*
//DFSPARM  DD *
VLSHRT
/*
//DUPS     DD SYSOUT=*
//PRIM     DD DISP=SHR,DSN=&&REPRO1
//         DD DISP=SHR,DSN=&&REPRO2
//RECS     DD DSN=&&RECS,REFDD=*.PRIM,SPACE=(CYL,(1,10),RLSE),
//          UNIT=SYSALLDA
//MERGED   DD DSN=&&REPROM,DISP=(,PASS),
//          REFDD=*.PRIM,SPACE=(CYL,(1,10),RLSE),
//          UNIT=SYSALLDA
//TOOLIN   DD *
  SORT FROM(PRIM) TO(RECS) USING(RECS)
  SELECT FROM(RECS) TO(MERGED) ON(5,44,CH) FIRST -
   DISCARD(DUPS)
//RECSCNTL DD *
  OPTION EQUALS
  SORT FIELDS=(5,44,CH,A)
/*
//*-------------------------------------------------------------------
//SOO3AMS  EXEC PGM=IDCAMS
//SYSPRINT  DD  SYSOUT=*
//MERGED    DD  DISP=SHR,DSN=&&REPROM
//SYSIN     DD  *
   DEFINE CLUSTER (NAME(New.Merged.OCDS) VOLUMES(vvvvvv) -
        CYLINDERS(nnn 0) -
        RECORDSIZE(1800 2040) FREESPACE(0 0) -
        INDEXED KEYS(44 0) SHAREOPTIONS(3 3) -
        SPEED BUFFERSPACE(530432) -
        UNIQUE NOWRITECHECK) -
        DATA(NAME(New.Merged.OCDS.DATA) -
        CONTROLINTERVALSIZE(12288)) -
        INDEX(NAME(New.Merged.OCDS.INDEX) -
        CONTROLINTERVALSIZE(2048))
   REPRO IFILE(MERGED) ODS(New.Merged.OCDS)
/*
```

16

DFSMSrmm considerations

This chapter discusses the following aspects of moving a system that is using Removable Media Manager (DFSMSrmm) into a sysplex where all the existing systems share a single DFSMSrmm database:

- Is it necessary to merge DFSMSrmm environments, or is it possible to have more than one DFSMSrmm environment in a single sysplex?
- A checklist of things to consider should you decide to merge the DFSMSrmms.
- A methodology for doing the merge.
- A list of tools and documentation that can assist with this exercise.

While some of the concepts in this chapter probably apply to tape management systems other than DFSMSrmm, this chapter has been written specifically for DFSMSrmm, by people familiar with that product. Correspondingly, the term we use in this chapter and in the index to describe the systems sharing a single tape management system database is RMMplex (as opposed to TAPEplex or TMSplex or something similar).

16.1 One or more RMMplexes

It *is* possible to have more than one RMMplex in a single sysplex. The definition of an RMMplex is the set of systems that share a single DFSMSrmm Control Data Set (CDS).

If your target environment is a BronzePlex, the incoming system may share tape drives with systems in the target sysplex; however, it will not share any DASD other than the volumes containing the CDSs. If it is not sharing catalogs and the DFSMSrmm database (at a minimum, and probably also the RACF database and SMS environment), then the incoming system would not be part of the same RMMplex as the target sysplex.

If your target environment is a GoldPlex, the incoming system would be sharing the system volumes (and therefore, potentially the volume containing the DFSMSrmm database) with the other systems in the target sysplex. However, because all the user catalogs and RACF and so on are *not* shared, you would once again probably not be merging your RMMplexes.

Finally, if your target environment is a PlatinumPlex, the incoming system *will* be sharing the user DASD (and RACF and SMS and so on) with the other systems in the target sysplex. In this case, you *would* normally merge the RMMplexes.

16.2 Considerations for merging RMMplexes

This section contains, in checklist format, a list of things that must be considered should you decide to merge two or more RMMplexes. We say RMMplexes rather than DFSMSrmm CDSs, because there are more things to be considered than simply merging the contents of the DFSMSrmm CDSs.

Due to the interrelationship between the following, it is strongly recommended that you merge the SMS environment, the HSM environment, the security environment (for example, RACF), the catalog environment, and the DASD environment, at the same time as merging the DFSMSrmm environments.

Table 16-1 Considerations for merging RMMplexes

Considerations	Note	Type	Done
Ensure toleration service is installed if necessary	1	P	
Identify and eliminate duplicate records in the DFSMSrmm CDSs	2	P	
Review Parmlib options	3	P	
Review DFSMSrmm Procedure	4	P	
Review DFSMSrmm CDS and Journal data set sizes, and attributes	5	P	
TCDB Tape CDS	6	P	
Review any housekeeping jobs	7	P	
Security Considerations	8	P	
Review DFSMSrmm exits	9	P	

The “Type” specified in Table 16-1 relates to the sysplex target environment—**B** represents a BronzePlex, **G** represents a GoldPlex, and **P** represents a PlatinumPlex. Notes indicated in Table 16-1 are described below:

1. Ensure toleration service is installed if necessary

Before you can start a DFSMSrmm merge process check the DFSMSrmm software levels. If the DFSMSrmm software levels are different, then you should check for compatibility PTFs, and confirm they are installed and applied.

To get a list of toleration and compatibility APARs for DFSMSrmm, refer to the DFSMS program directory, the DFSMSrmm PSP bucket, and Information APAR II08878.

2. Identify and eliminate duplicate records within the DFSMSrmm CDSs

Note: There are numerous references in this section to duplicate records and how to handle them. However, the first step has to be to *identify* if you have duplicate records. The easiest way to do this is to run the jobs in Example 16-1 on page 255 and Example 16-2 on page 255. The description of the first step of the job, STEP S001ICET on page 252, discusses how the merge job handles the various record types, and specifically, what it does with any duplicate records that it finds. You should look in the data sets containing the duplicate records to see which records types you need to address.

Figure 16-1 contains a graphiucal summary of the process of merging multiple RMM CDSs. In this example, each LPAR has its own CDS, whereas in our sysplex merge we will hopefully only have two—one from the incoming system, and one from the target sysplex.

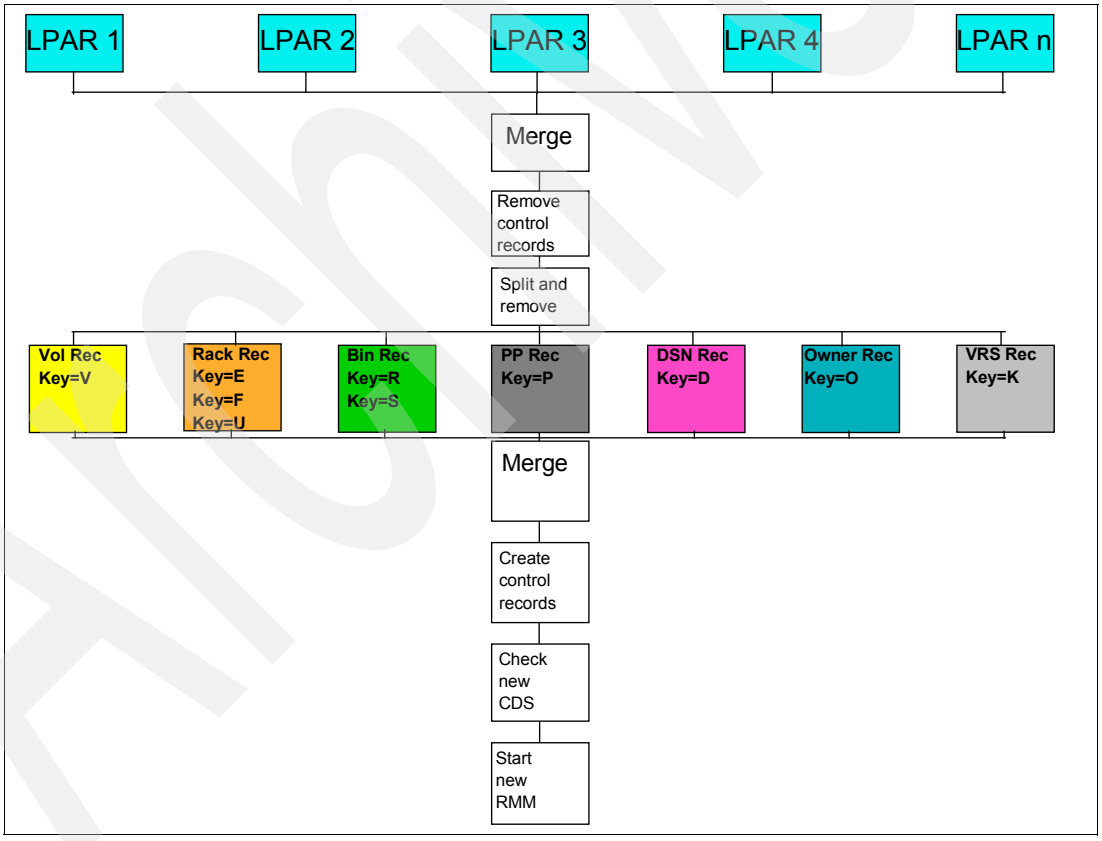


Figure 16-1 Merging process for multiple RMM CDSs

When merging CDSs, it is important to understand the contents of the CDS. For example, if you are merging two CDSs, you cannot merge them if the same resource is defined in each CDS. You must analyze the contents of the CDS to ensure that there are no duplicate resources (or at least, no conflicting duplicates).

If you find that there are a large number of duplicate tape volsers, then you should consider whether you really want to merge your DFSMSrmm CDSs. If you have a large number of duplicate volumes, and you would still like to merge the systems, then before you perform the merge, you should consider using one of the many TAPECOPY utility programs available, to copy the duplicate volumes to volumes with unique volsers.

There is a sample merge job shown in Example 16-2 on page 255. As well as performing the merge, the job can also be used to identify duplicate records in advance of the actual merge. The job can be continually rerun until all duplicates (or at least those that require action) have been resolved. The merge job is described in more detail in “16.3.1, “Merging the CDSs” on page 252.

The merge job repros the merging system's DFSMSrmm CDSs into a flat file and sorts the file, discarding duplicate records in the process, and creates a new merged DFSMSrmm CDS. Duplicate records are written to sysout files for you to investigate and act on.

Note that all the duplicates that are deleted are from the same system. If you plan to use the merge process to remove duplicate records, all of the records that you want to discard *must* come from the same system—for example, you can't use the merge to discard VRS records from RMMPLEX1 and Volume records from RMMPLEX2.

In this section we describe each resource type in the CDS and explain what you can do to handle duplicate records.

The record types contained in the DFSMSrmm CDS are:

Key type	Record type
C	Control
CA	Action
CM	Move
D	Data set
E	Empty rack
F	Scratch rack
K	VRS
O	Owner
P	Product
R	Empty bin
S	In-use bin
U	In-use rack
V	Volume

Volume record - The volume record contains detailed information about a volume and its contents. It also identifies the data sets that are on the volume, and volume chaining information. Some fields in the records identify the existence of other records in the CDS.

When moving volume records from one CDS to another, you must be sure to move the related records as well. These include the data set records, owner record, rack record, bin records, and product record. If you do not move them, the EDGUTIL utility may not be able to identify all the missing records and therefore may not be able to create all the missing records.

When you are merging CDSs, you should not have any duplicate volume serials. In fact, from an integrity point of view, you should not have duplicate volume serials in the same sysplex anyway, regardless of whether it causes a problem in DFSMSrmm or not. If you do, you must remove the duplicate volumes before starting the merge process. You should understand why there are duplicate volumes and, based on your understanding, select the

best approach for dealing with them. While the merge job will fix some duplicate records problems by just discarding one of the duplicates, you *must* ensure that there are no duplicate volumes records before you start the merge. When you run the merge job, duplicate volume records are written to the file specified on the DUPSMAIN DD statement.

If you have duplicate volumes, there could also be duplicate data set records for those volumes. If you delete the volumes, the related duplicate data set records will get deleted automatically as part of the process of removing the duplicate volume.

Control record - Each DFSMSrmm CDS contains a control record. The control record contains the dates and times of when the inventory management functions last ran, the counts of rack numbers and bin numbers for built-in storage locations, and the settings for some DFSMSrmm options.

When merging databases, only one control record is required. You must use the EDGUTIL program with PARM=MEND to correct the record counts after the merge job has completed, and before you start DFSMSrmm using the new CDS.

The control record also contains the CDSID value. When merging the databases, retain the control record from the system that contained the CDSID you wish to continue to use (probably the one from DFSMSrmm on the target sysplex). All systems that use the DFSMSrmm CDSs must have a matching CDSID specified in the EDGRMMxx member of Parmlib before being allowed to use the CDS.

As an alternative, you can bypass copying all control records, and after you have loaded the new DFSMSrmm CDS, you can create a new control record by using EDGUTIL with PARM=CREATE. After this has completed, use the same program with PARM=MEND to update the control record to correct the rack and bin counts.

You should use the RMM LISTCONTROL CNTL subcommand to check the options which are in use, such as EXTENDEDBIN and STACKEDVOLUME. If the options in use differ between the CDSs, we recommend that you enable options to bring the CDSs in line with each other and follow the steps to implement those options *before* starting the merge.

Action and Move records - Action records contain information about the librarian actions that are outstanding. They include the release actions and any volume movements that are required. Duplicate records will be deleted by the merge job, because there is a strong chance that they exist in each CDS, and the records will be rebuilt by DFSMSrmm during inventory management after the merge.

Data set record - A data set record exists for each volume on which the related data set has been written. Each record is uniquely identified by the volser and physical file number of the data set. Remember that it is perfectly acceptable to have multiple tapes all containing the same data set (unlike DASD data sets where you generally would try to avoid having the same data set name on more than one volume).

When merging CDSs, you want to be sure that there are no duplicate records across the two CDSs. Because the key of the data set record includes the volser, avoiding duplicate volsers ensures that there will be no duplicate data set records.

VRS record - The VRS records contain the retention and movement policies for your data. When merging the CDSs, you may find that you have the same policies defined in each CDS for some types of data. The policies are identified by fully qualified or generic data set name and job name, volser, and volser prefix.

When you merge CDSs, you must check for and resolve any duplicate policies. To help you do this, use the RMM SEARCHVRS TSO subcommand to get a list of all policies. If you have a policy for the same data set name and job name combination in more than one CDS, you should check that the retention and movement defined are the same. If they are the same, you can use the policy from any CDS. If they are not the same, you must resolve the conflict before merging the CDSs - that way duplicates can be ignored, regardless of which system the records are taken from.

The most important reason for checking the VRS definitions is to check which VRS policy will apply to a given data set/jobname combination, and what impact that could have after the merge. If you use generic VRS definitions (like TOTHSM.**), it is possible that the VRS definition controlling a given data set/job could change even if you don't have duplicate policy names, potentially resulting in you keeping fewer generations of that data set/job than you had intended. (This is similar to the discussion in Chapter 13, "RACF considerations" on page 189 about data sets being protected by different RACF profiles after a merge.) You should check all the VRS definitions for potential clashes between VRS definitions—if you find a clash, check, and, if necessary, modify the attributes of the VRS to meet your requirements.

There is a way to check whether the VRS records in the merged CDS will provide the results you expect. You can run EDGHSKP with PARM='VRSEL,VERIFY' using the test CDS that is created by the merge job. Specifying this PARM results in a trial run on inventory management. The resulting ACTIVITY file will contain a list of all changes that will be made. By using the EDGJACTP sample job and looking at the summary of changes in matching VRS, you can see if in fact you would have this problem.

Owner record - There is one owner record for each identified user that owns a DFSMSrmm resource such as a volume or VRS. There can also be owner records for users that own no resources. Perhaps you have defined these so that the information can be used within your installation.

The owner record contains detailed information about the user and the count of volumes owned. For users who own volumes, DFSMSrmm uses multiple owner records to provide directed retrieval of volume and data set information during search commands. You only need to copy the "Base" Owner records. When there are multiple owner records, there is no need to copy the Owner records that contain just volume information. Once the records are loaded into the new CDS, your merge job will use the EDGUTIL utility with the PARM=MEND option to set the correct counts of owned volumes and build the owner volume records for owned volumes. A sample is shown in Step S004MEND in the "DFSMSrmm Merge job" in Example 16-2 on page 255.

Duplicate owner records are not a problem if the duplicate owners are the same user. In this case, duplicates can be safely ignored, regardless of which system the records are taken from.

If the duplicate owners are different users, you must decide how to handle that situation. You may have to assign different user IDs or define a new owner ID and transfer the owned resources to the new owner ID. Your objective is to have no duplicate owners, or to have any duplicates be for the same user in each case.

To add a new owner, issue the following command on the RMMplex that the owner is being changed on:

```
RMM ADDOWNER new_owner_id DEPARTMENT(MERGE)
```

To delete the duplicate owner and transfer volume ownership to the new owner ID (after you have added the new owner ID), use the following command on the same RMMplex as the ADDOWNER command:

```
RMM DELETEOWNER old_owner_id NEWOWNER(new_owner-id
```

Product record - A product record identifies a single product and software level and the volumes associated with that product. When you are merging CDSs, the existence of duplicate product records can be resolved by running EDGUTIL MEND after the merge.

As an alternative to relying on MEND to resolve any duplicates, you can resolve the duplicates before the merge by choosing one of the product names to change. Firstly, list all the volumes associated with the product name, using a command similar to the following:

```
RMM LISTPRODUCT oldproduct
```

Then create a new product name using an ADDPRODUCT command, using a command similar to the following:

```
RMM ADDPRODUCT newproduct LEVEL(V01R01M00) NAME(.....) DESCRIPTION(.....)
OWNER(....)
```

Then change all the volumes that were associated with the old product name, to the new product name using a CHANGEVOLUME command, as follows:

```
RMM CHANGEVOLUME vvvvvv NUMBER(newproduct) FEATCD(A001) LEVEL(V01R03M00)
```

Finally you should delete the old product name using a DELETEPRODUCT command, as follows:

```
RMM DELETEPRODUCT oldproduct LEVEL(V01R01M00)
```

If you do not resolve the duplicates before the merge, there will be duplicate product records during the merge, and all the ones in the second CDS specified on the EXTRACT DD statement will be discarded during the merge—the discarded records are written to the file specified on the DUPSMAIN DD statement. Therefore, if you decide to leave some duplicate records in the CDS so they can be deleted during the merge, you must make sure that all those records are in the same CDS. After the merge, running EDGUTIL MEND will clean up the associations between the Product and Volume records.

Rack numbers record - There is a single rack record for each shelf location you have defined to DFSMSrmm. Rack records are optional when the rack number and the volume serial number are the same value. There may be one for each volume defined, but there can be more if you also defined empty rack numbers for use when volumes are added to the CDS. The rack number represents the volume's external volser.

The DFSMSrmm record for each rack number can begin with one of three keys - E for empty, F for full, and U for in-use. So, it is possible that you could have duplicate rack numbers in the two CDSs without the merge job spotting them as duplicates. For example, rack number 200000 in one DFSMSrmm CDS could be Empty (so the key would be E200000), and rack number 200000 in the ther DFSMSrmm CDS could be Full (so the key would be F200000)—however, the merge job ignores the key when identifying duplicate records, so duplicate rack numbers *will* be identified.

When you merge CDSs, you must not have any duplicate rack numbers. Any duplicates must be deleted prior to the merge. Note that if there are duplicate records, the “U” records will be selected ahead of the “F” records, and the “E” records will be selected last. Duplicates will be discarded, regardless of which CDS each record may have resided in prior to the merge. Any duplicates found at the time of the merge will be written to the file specified on the DUPSEFU DD statement.

Each rack number must have only one record type—either E, F, or U; you cannot have two record types for an individual rack number. If the duplicate already contains a volume, you can reassign the volume to another rack number by using the RMM CHANGEVOLUME subcommand. There are two different ways of reassigning a volume to a new rack number:

```
RMM CHANGEVOLUME PP1234 POOL(A*)       or,
RMM CHANGEVOLUME PP1234 RACK(AP1234)
```

In the first command, DFSMSrmm picks an empty rack in pool A*. In the second example, you have selected the empty rack to be used. Remember that the rack number is the external label on the tape, so changing large numbers of rack numbers is not a trivial exercise. It helps if you can select a new rack number that is similar to the old one (for example, we changed rack number PP1234 to rack number AP1234). Before cleaning up any duplicate rack numbers, refer to the description of the Volume record on page 244 for an explanation of how to handle duplicate volumes.

If you currently have rack numbers for volumes when the rack number and the volume serial number are the same value, you could choose to stop using rack numbers before starting the merge. This would reduce the numbers of records in the CDS. To change a volume to no longer use a rack number you can use the following commands:

```
RMM CHANGEVOLUME 100200 NORACK
RMM DELETERACK 100200
```

Bin numbers record - There is a single bin record for each storage location shelf location you have defined to DFSMSrmm.

As with rack numbers, each bin number record can potentially have one of two different keys; in this case, R (empty) and S (in-use). Because the same bin can be represented by two different keys, the merge jobs may not correctly identify all duplicates. For example, bin number 000029 could be represented by record R000029 in one CDS and by record S000029 in the other. However, as with the rack numbers, the sample merge job is set up to ignore the key when searching for duplicate bin records, so duplicates will be identified (they are written to the file specified on the DUPSRS DD statement). Note that if there are duplicate records, the “S” records will be selected and the “R” records discarded, regardless of which CDS each record may have resided in prior to the merge.

When you merge CDSs, you must not have any duplicate bin numbers. Any duplicates must be addressed before you do the merge. If the duplicate already contains a volume, you can reassign the volume to another bin number by using the RMM CHANGEVOLUME subcommand. You can reassign a volume to a new bin number by issuing the following command:

```
RMM CHANGEVOLUME A00123 BIN(000029)
```

You can also remove the volume from the storage location, free up the bin number, and delete the unused bin number. The following commands show how to delete duplicate bin number X00022 in location VAULT1 of media name CARTS:

```
RMM CHANGEVOLUME A00123 LOCATION(HOME) CMOVE
RMM DELETEBIN X00022 MEDIANAME(CARTS) LOCATION(VAULT1)
```

The next time you run inventory management, the volume is reassigned to the storage location, and another bin number is assigned.

Note: When you need to reassign bin numbers, you should consider how off-site movements are managed in your installation. If you depend on the volume being in a specific slot in a storage location, you should ensure that any volumes you change are physically moved in the storage location.

3. Review DFSMSrmm Parmlib options

   To allow for a simplified backout process, it is recommended that you do not change the current DFSMSrmm Parmlib member, but rather create a new merged version. On the actual merge day, you can simply rename the members to pick up the merged definitions.

   You should review the IEFSSNxx member, and confirm the DFSMSrmm subsystem name and definitions are the same on all systems that will be sharing the same DFSMSrmm CDS.

   You should review the EDGRMMxx member for differences, and consolidate the members. Be aware of any changes you make that require supporting changes to RACF. You should also investigate whether all systems can share the same EDGRMMxx Parmlib member as opposed to having a separate member for each DFSMSrmm instance.

   The EDGRMMxx options that are affected by the merging of CDSs are:

   – OPTIONS
   – LOCDEF
   – VLPOOL

– REJECT

OPTIONS - The options identify the installation options for DFSMSrmm. When you are merging CDSs, consolidate the OPTIONS command operands from all CDSs you are merging to ensure that you are using the same OPTIONS command operands on all DFSMSrmm subsystems.

After you merge DFSMSrmm CDSs, the CDSID of at least one DFSMSrmm subsystem will probably change. You must update the CDSID parameter to match the new CDSID, or if you have created a new DFSMSrmm CDS control record and you have not specified a CDSID, DFSMSrmm creates the ID in the control record from the CDSID automatically.

If the DFSMSrmm CDSs are kept in synch with the catalogs on the system, and you have enabled this using the inventory management CATSYNCH parameter, you should review the CATSYSID values specified for each system.

If all systems sharing the merged DFSMSrmm CDS will also have access to the same user catalogs for tape data sets (which should be the case), you can use CATSYSID(*). However, if the catalogs are not fully shared, you must produce a consolidated list of systems IDs for use with the CATSYSID and set the correct values in EDGRMMxx for each system.

Before you use the merged CDS, we recommend that you mark the CDS as not in synch with the catalogs. Use the EDGUTIL program with UPDATE and the SYSIN statements with CATSYNCH(NO) to do this. Once you have started DFSMSrmm using the newly merged CDS, you can re-enable catalog synchronization using inventory management as detailed in the *z/OS DFSMSrmm Implementation and Customization Guide,* SC26-7405.

If you use the VRSMIN option to specify a minimum number of VRSs, remember to update the value if you have a different number as a result of the merge.

LOCDEF - These options identify your locations to DFSMSrmm. When you are merging CDSs, consolidate the LOCDEF options from all CDSs you are merging to ensure that you have no duplicates and that no locations are left out. If you have duplicates, ensure that they are identified as the same location type and with the same media names.

VLPOOL - These options identify the ranges of rack numbers and pools of shelf space you have in your library. If your library is not changing, we recommend that you do not change your current VLPOOL options.

If you are merging CDSs and your existing VLPOOL definitions are different for each existing CDS, you must merge the definitions together to cover the merged sets of volumes and rack numbers. Ensure that the operands specified on each VLPOOL are correct when the same pool prefix is defined in more than one source environment.

REJECT - These options identify the ranges of volumes to be rejected for use on the system. The reject specifies the rack number prefix or, if a volume has no library shelf location, DFSMSrmm checks the volume serial number. The range of volumes you specify on the REJECT statement should match the VLPOOL definitions that you have defined in RMM. For example, if you currently specify that the incoming system can only use volumes beginning with A*, and the target sysplex systems can only use volumes beginning with B*, you will need to update the REJECT statement at the time of the merge to indicate that all those systems can use tapes starting with either A* *or* B*.

Where you have a tape library shared by multiple systems, define the same VLPOOL definitions to all systems and use the REJECT definitions to prevent systems from using volumes that are for use on other systems.

The actions you take for the REJECT definitions should match how you handle the VLPOOL definitions.

The REJECT options can be used, for volumes defined to DFSMSrmm, to control the use of volumes on the system. And for undefined volumes, it can be used to control the partitioning of system managed libraries. Consider both of these aspects as you merge the REJECT definitions.

4. Review the DFSMSrmm started task JCL

   To allow for a simplified backout process, it is recommended that you do not change the current DFSMSrmm procedure, but rather create a new merged version. On the actual merge day, you can simply rename the two procedures to perform the merge.

   You should review the DFSMSrmm procedure and investigate whether all systems can share the same DFSMSrmm Procedure. This should be possible with little or no work as the only system-specific information that is contained in the DFSMSrmm procedure is the names of the PDO (EDGPDOx) data sets. PDO data sets are optional, but if you do define them, they should be unique on each system—we suggest that you include the system name (&SYSNAME) in the data set name.

   The DFSMSrmm CDS and Journal names can be specified in either the DFSMSrmm JCL or in the Parmlib member EDGRMMxx. For flexibility, we recommend that they are specified in EDGRMMxx; this allows you to easily change the CDS name by switching to a different Parmlib member.

   You should also check that the REGION size specified for the DFSMSrmm started procedure is large enough; review the latest recommendations for this in the *z/OS DFSMSrmm Implementation and Customization Guide,* SC26-7405.

5. Review the DFSMSrmm CDS and Journal data set sizes and attributes

   To allow for a simplified backout process, we recommend that you create new versions of the DFSMSrmm CDS and Journal data sets. On the actual merge day, you can simply rename the old ones, and rename the new ones to be ready to start DFSMSrmm after the merge**.**

   Before you create the new data sets, calculate the new combined CDS size, and update the merge job with this information so that the CDS allocated in that job is valid.

   At the same time, calculate the new Journal data set size, and create a job to allocate the data set.

   Review the sample allocation jobs (for the DFSMSrmm CDS, member EDGJMFAL in SYS1.SAMPLIB, and for the DFSMSrmm Journal, member EDGJNLAL in SYS1.SAMPLIB) to ensure that you are defining the new CDS with the latest information regarding record size, CISize, Shareoptions, and so on. For more information about these data sets, refer to *z/OS DFSMSrmm Implementation and Customization Guide,* SC26-7405.

6. TCDB Tape Configuration data set

   If the systems to be merged contain Volume Entries cataloged inside either a General Tape Volume Catalog (SYS1.VOLCAT.VGENERAL, for example) or a specific Tape Volume Catalog (SYS1.VOLCAT.Vx, for example), you need to list all the volume entries and create the volume entries in the target TCDB with the same attributes as in the current system. You can use a REPRO MERGECAT to copy entries from the old to the new volume catalogs if required.

   You will also need to confirm that the SMS ACS routines are allocating tape data sets to the appropriate Storage Group(s) associated with the Tape Library(s).

   Review the DEVSUPxx member in SYS1.PARMLIB if the TCDB is being merged and make sure that all systems are entering the tapes with the same category number.

If the category number changes for any systems, there is no need to make any changes for private volumes since when the volume returns to scratch, its category will be changed to one of the new scratch categories. However, you will need to make changes for volumes already in Scratch status. There is more information about this in the section entitled "Processing Default Categories when using DEVSUPxx in an ATLDS" in *z/OS DFSMS Object Access Method Planning, Installation, and Storage Administration Guide for Tape Libraries,* SC35-0427.

You will need to obtain a list of volumes whose storage group name is *SCRTCH* using ISMF Option 2.3 Mountable Tape. Use the ISMF ALTER command (not the line operator) to change the volume use attribute for all volumes in the list from scratch to scratch; this causes the library manager category for each volume to be changed to the new value established through DEVSUPxx.

7. Review any housekeeping jobs

   Review these jobs, confirming that data set names are valid for the new merged environment. It should be possible to eliminate any DFSMSrmm jobs that were running on the incoming system previously, except for any report jobs that were not running on the target sysplex. If you have switched on the catalog synchronization feature, you must check that the CATSYSID definitions are correct and that the inventory management vital record processing is running on the correct system. Also, the inventory management expiration processing must be checked and running on multiple systems if not all user catalogs are shared (hopefully, in a PlatinumPlex, *all* catalogs are shared between all systems).

8. Security considerations

   Chapter 8 in the *z/OS DFSMSrmm Implementation and Customization Guide,* SC26-7405 discusses security for DFSMSrmm. If the systems to be merged are also merging the RACF environment at the same time, and you have set up the Parmlib definitions the same on all systems, there should be no additional security concerns. If TAPEVOL and/or TAPEDSN are active, make sure all EDGRMMxx definitions for TPRACF are the same. If you have not switched on TAPEVOL and/or TAPEDSN on all systems before you have merged the DFSMSrmm CDS, but now you will use them on all systems sharing the DFSMSrmm CDS, you must define all needed resources before the merged DFSMSrmm is started.

9. Review DFSMSrmm exits

   You will need to determine what DFSMSrmm exits are in use (for example, EDGUX100, EDGUX200, CBRUXCUA, CBRUXENT, CBRUXEJC and CBRUXVNL). You should review the functions of these exits on the incoming system and the target sysplex systems. If they are not the same, you should see if you can create a single version of each exit so that all systems will be using identical exits, this ensuring consistent processing.

16.3 Methodology for merging RMMplexes

In this section we discuss the various approaches that can be taken to merging RMMplexes, and give some advice about the approach that provides the smallest risk or is the easiest to implement.

The approach discussed in this Redbook uses REPRO to reproduce the merging system's DFSMSrmm CDSs, sorts the resulting flat file, and creates a new merged DFSMSrmm CDS. Duplicate and unnecessary records are not included in the new CDS. Duplicates are written to sysout files for your review, and to confirm that you do not require them. DFSMSrmm is then restarted on all systems, using the new merged DFSMSrmm CDS.

An alternate approach is to create DFSMSrmm ADD....commands for all volumes, data sets, racks, bins and vital record specifications being merged. This approach can be used to avoid any outage to DFSMSrmm. However, care must be taken to ensure that the appropriate security options are already in place; for example, TAPEVOL/TAPEDSN(TVTOC). As there are no tools available to do this, you would need to create these yourself.

16.3.1 Merging the CDSs

Follow these steps to merge the CDSs:

1. Prior to the day of the merge, you should have addressed all the issues and tasks identified in 16.2, “Considerations for merging RMMplexes” on page 242.
2. Stop DFSMSrmm on all systems in both RMMplexes. When you run the merge job, DFSMSrmm on all systems must be down. When you stop DFSMSrmm, do *not* use the OPT=RESET function, as it allows tapes to be processed without DFSMSrmm's knowledge. You will not be able to use any tapes from this time until the merge is complete.
3. Back up both sets of CDSs and Journals using the sample JCL in Example 16-1 on page 255. This JCL uses Repro to back up the files. While you should not change any of these files during the merge, you should take the backups to protect yourself from any accidental damage. Also, by using Repro for the backup, you can then use the backup files as input to the merge process.
4. If required, perform any merging of Volume Entries in the Target TCDB Tape Catalog Database.
5. Run the merge job shown in Example 16-2 on page 255, using the data sets created in Step 3.

   The sample merge job uses ICETOOL. If you do not have this program available, an alternate method using IDCAMS is discussed in the IBM Redbooks *Converting to RMM - A Practical Guide,* SG24-4998 and *DFSMSrmm Primer,* SG24-5983 (in the chapter entitled “Merging and Splitting the DFSMSrmm CDS”). If SYNCTOOL is available instead of ICETOOL, this job may run with little or no changes to the JCL and statements.

   The sample merge job can be run while RMM is running while you are doing testing. However, when you are doing the actual merge, all RMM subsystems should be stopped before you run the merge job.

   The merge job consists of a number of steps. The steps, and the data sets used by each step, are described here:

   Step CLEANUP

   The first step in the job simply deletes the output files that may have been left from the last run of this job.

   Step S001ICET

   The next step in the job uses the ICETOOL utility of DFSORT. The input data sets are sequential unloads of the two DFSMSrmm CDSs (created with Repro) and are specified on the DD EXTRACT statement. If duplicates are discovered in the EXTRACT files, the first occurrence will be retained and the second and subsequent ones will be discarded (with the exceptions previously noted for the rack and bin records).

   Therefore, the order of the DD statements on the EXTRACT is critical. The first DD should refer to the flat file from the RMMplex whose CDSID you wish to retain. Whenever duplicate records are detected, the records from that RMMplex will be retained and those from the other RMMplex will be discarded. For this reason, you must be sure that any records that you want the merge job to discard are all in the same RMMplex.

The MAINCNTL DD statement contains the ICETOOL statements that will be used to sort and select most record types (C, K, P, and V). The associated SELECT statement in the TOOLIN DD specifies that duplicates should not be written to the output file for these record types (MAIN1S), but should instead be written to DUPSMAIN for you to investigate and resolve.

The EFUXCNTL DD statement contains the ICETOOL statements that will be used to sort and select record types E, F, and U. They also ensure that the output file (EFU1S) will have only one record type for each rack. If the same rack number is discovered in more than one record type, records will be selected with the “U” types first, followed by the “F” types, and finally the “E” types. Duplicate records will be written to DUPSEFU for you to investigate and resolve.

The RSXXCNTL DD statement contains the ICETOOL statements that will be used to sort and select record types “R” and “S” into the data set specified on the RS1S DD statement. They also make sure you have only one record type for each bin. If the same bin number is discovered in more than one record type, the “S” record will be selected, and the “R” record will be written to DUPSRS for you to investigate and resolve.

The OOOOCNTL DD statement contains the ICETOOL statements that will be used to sort and select the type O records into the data set specified on the OO1S DD statement and to make sure you have only one record for each owner. It will also discard type O records that just contain volume information (refer to the owner record description on page 246).

Finally, the DDDDCNTL DD statement contains the ICETOOL statements that will be used to sort and select all the data set records. If you have addressed any duplicate volumes, there should be no issues with copying all data set records, as the key includes the volser and the physical file number; therefore, we have not provided a DUPSD DD statement as a destination for duplicate records.

Step S002ICET

This step also uses the ICETOOL Utility of DFSORT. The input data sets are the temporary data set created by S001ICET that contain the (non-duplicate) CDS records. An output data set (DD MERGED) will be created in the same format as the input data sets for S001ICET.

Step S003AMS

This step uses IDCAMS to define a new DFSMSrmm CDS, and Repro' the sequential file created in S002ICET (DD MERGED) into the new CDS—this is the CDS that will be used in the new merged RMMplex.

Step S004MEND

This step uses the EDGUTIL utility to repair all the record counts, set correct owners, and create missing records, as described previously. Running EDGUTIL with the MEND parameter only updates the CDS referred to on the MASTER DD card, so you can safely run this utility as many times as you like prior to the actual merge, *as long as you ensure that the MASTER DD points at your test CDS*.

Note that if any of the volumes defined in the newly merged DFSMSrmm CDS are contained within an ATL or VTS, then the TCDB will need to already contain correct entries for these volumes—that is, SYS1.VOLCAT.VGENERAL will need to already have these volumes defined before the MEND can complete successfully. If all the volumes are inside an ATL or VTS, you can use the EDGUTIL MEND(SMSTAPE) function to add/correct the volumes in the TCDB. **Note that if you specify SMSTAPE, the TCDB will be updated, so you should not use this option until you do the actual merge.**

Step S005MEND

Note that, if there are many records to be fixed, there can be quite a lot of output from EDGUTIL. In order to get a clear picture of which records the MEND was unable to fix (and there should be *very* few of these), you should run the EDGUTIL program again, this time with the VERIFY parameter. This will identify any actions that MEND was not able to take, and therefore the records you need to consider taking action for. The messages you are most likely to see are "VOLUME xxxxxx NOT FOUND IN VOLUME CATALOG", and these will be fixed when you run MEND with the SMSTAPE option.

6. Run EDGUTIL with UPDATE to turn off CATSYNCH. It is neceassary to turn off catalog synchronization before the merge is complete as the catalogs will need to be re-synchronized. Use the sample job in Example 16-3 on page 258.
7. Perform any renames of CDSs, Journals, Parmlib members, started task JCL and so on.
8. Start the DFSMSrmm procedure, using the new Parmlib member.
9. Check that the merged information is valid and correct by using DFSMSrmm commands and the ISPF dialog to retrieve and display information.
10. Run EDGHSKP with CATSYNCH if catalog synchronization is to be maintained by DFSMSrmm. We recommend doing this for the performance benefits it can have for inventory management when use is made of policies on catalog retention. This should be done before the first housekeeping job (EDGHSKP) is run after the merge.
11. If re-synchronization between the new merged DFSMSrmm CDS and TCDB is required, the DFSMSrmm EDGUTIL program should be run with a parm of 'VERIFY(VOLCAT)'. In addition, if the systems in the RMMplex are all running OS/390 2.10 or later, you should also specify 'VERIFY(SMSTAPE)'.
12. Run DFSMSrmm inventory management on the new CDS. During this run, DFSMSrmm VRSEL and DSTORE processing identifies volumes to be moved to the correct location. If any moves are identified, use EDGRPTD to produce the picking lists and actions of any moves that you know are still required.

    The inventory management run should execute all inventory management functions, including backup.

16.3.2 Backout

If backout is required at any stage, because you created new versions of the CDS, Journal, Parmlib member and started task JCL, you can simply rename the old data sets back to the original names. There should be no problems as a result of the additional Volume Entries you may have created in the TCDB; however, they will need to be cleaned up before re-attempting the merge process.

16.4 Tools and documentation

In this section we provide information about tools and documentation that may help you complete this task.

16.4.1 Tools

Example 16-1 contains sample JCL that can be used to back up the DFSMSrmm CDSs and Journal data sets after the DFSMSrmm subsystems have been stopped. Note that this job will *not* reset the contents of the Journal data sets. Even though we will not be making any changes to any of these data sets during the merge process, it is prudent to take a backup of them before any changes are made, simply to protect yourself from any mistakes that might arise during the merge.

You can also use this JCL to create the input to the merge job for dummy runs, in order to identify any duplicate records that may exist. Note, however, that if you run this JCL while the DFSMSrmm subsystems are running, then you must comment out the //MASTER and //JOURNAL DD cards. If DFSMSrmm is *not* running when you run the EDGBKUP job, you will need to uncomment the MASTER and JOURNAL DD statements and fill in the correct names for those data sets.

Note: During the dummy runs, while you are testing the merge jobs and identifying duplicate records, the jobs can be run while the RMM subsystem is up and running. However, when you are ready to do the actual merge, the RMM subsystem should be stopped before the job is run.

Example 16-1 JCL to Back up the DFSMSrmm CDSs and Journals

```
//S001BKUP EXEC PGM=EDGBKUP,PARM='BACKUP(NREORG)'
//SYSPRINT  DD  SYSOUT=*
//*MASTER    DD  DISP=SHR,DSN=RMMPLEX1.RMM.CDS
//*JOURNAL   DD  DISP=SHR,DSN=RMMPLEX1.RMM.JOURNAL
//BACKUP    DD  DSN=hlq.RMMPLEX1.CDS.BACKUP,DISP=(,CATLG,DELETE),
//           DCB=(DSORG=PS,RECFM=VB,LRECL=9216),
//           UNIT=SYSALLDA,SPACE=(CYL,(xxx,xx),RLSE)
//JRNLBKUP  DD  DSN=hlq.RMMPLEX1.JRNL.BACKUP,DISP=(,CATLG,DELETE),
//           DCB=(DSORG=PS,RECFM=VB,LRECL=32756,BLKSIZE=32760),
//           UNIT=SYSALLDA,SPACE=(CYL,(xxx,xx),RLSE)
//S002BKUP EXEC PGM=EDGBKUP,PARM='BACKUP(NREORG)'
//SYSPRINT  DD  SYSOUT=*
//*MASTER    DD  DISP=SHR,DSN=RMMPLEX2.RMM.CDS
//*JOURNAL   DD  DISP=SHR,DSN=RMMPLEX2.RMM.JOURNAL
//BACKUP    DD  DSN=hlq.RMMPLEX2.CDS.BACKUP,DISP=(,CATLG,DELETE),
//           DCB=(DSORG=PS,RECFM=VB,LRECL=9216),
//           UNIT=SYSALLDA,SPACE=(CYL,(xxx,xx),RLSE)
//JRNLBKUP  DD  DSN=hlq.RMMPLEX2.JRNL.BACKUP,DISP=(,CATLG,DELETE),
//           DCB=(DSORG=PS,RECFM=VB,LRECL=32756,BLKSIZE=32760),
//           UNIT=SYSALLDA,SPACE=(CYL,(xxx,xx),RLSE)
```

Example 16-2 contains the merge job that we have referred to and discussed earlier in this chapter. This job can also be used to identify duplicate records during the preparation for the merge.

Example 16-2 DFSMSrmm Merge Job

```
//CLEANUP EXEC PGM=IEFBR14
//DD1      DD DSN=hlq.MERGED.RMM.MASTER.SEQ,DISP=(MOD,DELETE),
//         SPACE=(CYL,0),UNIT=SYSDA
//DD2      DD DSN=hlq.MERGED.RMM.MASTER,DISP=(MOD,DELETE),
//         SPACE=(CYL,0),UNIT=SYSDA
/*
//S001ICET EXEC  PGM=ICETOOL
```

```
//TOOLMSG  DD SYSOUT=*
//DFSMSG   DD SYSOUT=*
//DFSPARM  DD *
VLSHRT
/*
//DUPSMAIN DD SYSOUT=*
//DUPSEFU  DD SYSOUT=*
//DUPSRS   DD SYSOUT=*
//DUPSO    DD SYSOUT=*
//EXTRACT  DD DISP=SHR,DSN=hlq.RMMPLEX1.CDS.BACKUP
//         DD DISP=SHR,DSN=hlq.RMMPLEX2.CDS.BACKUP
//MAIN     DD DSN=&&MAIN,REFDD=*.EXTRACT,SPACE=(CYL,(1,10),RLSE),
//          UNIT=3390,DISP=(,PASS)
//EFU      DD DSN=&&EFU,REFDD=*.EXTRACT,SPACE=(CYL,(1,10),RLSE),
//          UNIT=3390,DISP=(,PASS)
//RS       DD DSN=&&RS,REFDD=*.EXTRACT,SPACE=(CYL,(1,10),RLSE),
//          UNIT=3390,DISP=(,PASS)
//OOOO     DD DSN=&&OOOO,REFDD=*.EXTRACT,SPACE=(CYL,(1,10),RLSE),
//          UNIT=3390,DISP=(,PASS)
//DDDD     DD DSN=&&DDDD,REFDD=*.EXTRACT,SPACE=(CYL,(1,10),RLSE),
//          UNIT=3390,DISP=(,PASS)
//MAIN1S   DD DSN=&&MAIN1S,REFDD=*.EXTRACT,SPACE=(CYL,(1,10),RLSE),
//          UNIT=3390,DISP=(,PASS)
//EFU1S    DD DSN=&&EFU1S,REFDD=*.EXTRACT,SPACE=(CYL,(1,10),RLSE),
//          UNIT=3390,DISP=(,PASS)
//RS1S     DD DSN=&&RS1S,REFDD=*.EXTRACT,SPACE=(CYL,(1,10),RLSE),
//          UNIT=3390,DISP=(,PASS)
//OOO1S    DD DSN=&&OOO1S,REFDD=*.EXTRACT,SPACE=(CYL,(1,10),RLSE),
//          UNIT=3390,DISP=(,PASS)
//TOOLIN   DD *
  SORT FROM(EXTRACT) TO(MAIN) USING(MAIN)
  SELECT FROM(MAIN) TO(MAIN1S) ON(5,44,CH) FIRST -
   DISCARD(DUPSMAIN)
  SORT FROM(EXTRACT) TO(EFU) USING(EFUX)
  SELECT FROM(EFU) TO(EFU1S) ON(15,6,CH) FIRST -
   DISCARD(DUPSEFU)
  SORT FROM(EXTRACT) TO(RS) USING(RSXX)
  SELECT FROM(RS) TO(RS1S) ON(6,23,CH) FIRST -
   DISCARD(DUPSRS)
  SORT FROM(EXTRACT) TO(OOOO) USING(OOOO)
  SELECT FROM(OOOO) TO(OOO1S) ON(5,44,CH) FIRST -
   DISCARD(DUPSO)
  SORT FROM(EXTRACT) TO(DDDD) USING(DDDD)
//MAINCNTL DD *
  OPTION EQUALS
  SORT FIELDS=(5,44,CH,A)
  INCLUDE COND=(5,2,CH,NE,C'CA',AND,5,2,CH,NE,C'CM',AND,
                5,1,CH,NE,C'O',AND,
                5,1,CH,NE,C'D',AND,
                5,1,CH,NE,C'R',AND,5,1,CH,NE,C'S',AND,
                5,1,CH,NE,C'E',AND,5,1,CH,NE,C'F',AND,5,1,CH,NE,C'U')
/*
//EFUXCNTL DD *
  OPTION EQUALS
  SORT FIELDS=(5,1,CH,D,15,6,CH,A)
  INCLUDE COND=(5,1,CH,EQ,C'E',OR,5,1,CH,EQ,C'F',OR,5,1,CH,EQ,C'U')
/*
//RSXXCNTL DD *
  OPTION EQUALS
  SORT FIELDS=(6,23,CH,A,5,1,CH,D)
```

```
  INCLUDE COND=(5,1,CH,EQ,C'R',OR,5,1,CH,EQ,C'S')
/*
//OOOOCNTL DD *
  OPTION EQUALS
  SORT FIELDS=(5,1,CH,A)
  INCLUDE COND=(5,1,CH,EQ,C'O',AND,14,1,CH,EQ,X'OO')
/*
//DDDDCNTL DD *
  OPTION EQUALS
  SORT FIELDS=(5,1,CH,A)
  INCLUDE COND=(5,1,CH,EQ,C'D')
/*
//SOO2ICET EXEC  PGM=ICETOOL
//TOOLMSG  DD SYSOUT=*
//DFSMSG   DD SYSOUT=*
//DFSPARM  DD *
VLSHRT
/*
//EXTRACT  DD DISP=SHR,DSN=&&MAIN1S
//         DD DISP=SHR,DSN=&&EFU1S
//         DD DISP=SHR,DSN=&&RS1S
//         DD DISP=SHR,DSN=&&OOO1S
//         DD DISP=SHR,DSN=&&DDDD
//MERGED   DD DSN=hlq.Merged.RMM.Master.Seq,
//          REFDD=*.EXTRACT,SPACE=(CYL,(xxx,yy),RLSE),
//          UNIT=SYSALLDA,DISP=(,CATLG)
//TOOLIN   DD *
  SORT FROM(EXTRACT) TO(MERGED) USING(MOST)
//MOSTCNTL DD *
  OPTION EQUALS
  SORT FIELDS=(5,56,CH,A)
/*
//SOO3AMS  EXEC PGM=IDCAMS,REGION=OM
//SYSPRINT  DD  SYSOUT=*,OUTLIM=1OOOOOO
//RMM       DD  DISP=SHR,DSN=hlq.Merged.RMM.Master.Seq
//SYSIN     DD  *
  DEFINE CLUSTER(NAME(hlq.Merged.RMM.Master) -
                 FREESPACE(2O 2O) -
                 KEYS(56 O) -
                 REUSE -
                 RECSZ(512 9216) -
                 SHR(3 3) -
                 CYLINDERS(xxx yy) -
                 VOLUMES(vvvvvv)) -
           DATA(NAME(hlq.Merged.RMM.Master.DATA) -
                 CISZ(1O24O)) -
          INDEX(NAME(hlq.Merged.RMM.Master.INDEX) -
                 CISZ(2O48) -
                 NOIMBED -
                 NOREPLICATE)
   REPRO IFILE(RMM) ODS(hlq.Merged.RMM.Master)
/*
//*-------------------------------------------------------------------
//SOO4MEND  EXEC PGM=EDGUTIL,PARM='MEND'
//SYSPRINT  DD   SYSOUT=*
//MASTER    DD   DISP=SHR,DSN=hlq.Merged.RMM.Master
//SYSIN     DD   DUMMY
//*-------------------------------------------------------------------
//SOO5MEND  EXEC PGM=EDGUTIL,PARM='VERIFY'
//SYSPRINT  DD   SYSOUT=*
```

```
//MASTER    DD   DISP=SHR,DSN=hlq.Merged.RMM.Master
//SYSIN     DD   DUMMY
```

Example 16-3 contains sample JCL that can be used to turn the CATSYNCH feature off after you have merged the CDSs and before you start the DFSMSrmm subsystems using the new CDS.

Example 16-3 CATSYNCH Job

```
//EDGUTIL  EXEC PGM=EDGUTIL,PARM='UPDATE'
//SYSPRINT  DD  SYSOUT=*
//MASTER    DD  DISP=SHR,DSN=hlq.Merged.RMM.Master
//SYSIN     DD  *
  CONTROL CDSID(Merged) CATSYNCH(NO)
/*
```

16.4.2 Documentation

Finally, the following documents contain information that will be helpful as you proceed through this project:

- *z/OS DFSMSrmm Implementation and Customization Guide,* SC26-7405
- *z/OS DFSMSrmm Reporting,* SC26-7406
- *Converting to RMM - A Practical Guide,* SG24-4998
- *DFSMSrmm Primer,* SG24-5983

17

OPC considerations

Nearly all OS/390 and z/OS customers use some batch scheduling product to control their production batch work. IBM's offering in this area is called Operations Planning and Control (OPC). This chapter discusses the following aspects of moving a system which has its own OPC controller into a sysplex that is currently using its own OPC controller:

- Is it necessary to merge OPC environments, or is it possible to have more than one OPC environment in a single sysplex?
- A checklist of things to consider should you decide to merge the OPCs.
- A methodology for doing the merge.
- A list of tools and documentation that can assist with this exercise.

While some aspects of the contents of this chapter may also apply to other scheduling products, this chapter was written specifically for OPC, by specialists with extensive OPC experience, and therefore cannot be considered a generic guide for merging job scheduling products.

Note: OPC has been superseded by a product called Tivoli Workload Scheduler (TWS) V8. However, the discussion and examples provided in this chapter apply equally to OPC and TWS, so we use the term OPC to refer to both OPC and TWS.

17.1 One or more OPCplexes

An OPCplex consists of one OPC controller address space which schedules all work, and multiple tracker address spaces which track the work across a number of systems. A tracker is required for each and every system in the OPCplex. However, there is only one active controller at any given time. You may have other standby controllers in the OPCplex; the standby controllers can take over control for scheduling the work should the active controller fail.

It *is* possible to have more than one OPCplex in a sysplex. Each OPCplex has its own databases, which are owned by the controller and its own trackers, which connect to the controller. It is also possible for the OPCplex to extend *beyond* the sysplex.

The connection between the controller and each tracker can be either:

- Shared DASD
- VTAM
- XCF (if in a monoplex or multisystem sysplex)

When they are in the same sysplex, the controller and tracker should be connected using XCF as this is the simplest connection type, and also allows OPC to utilize XCF facilities to have a standby controller. If the OPCplex needs to extend beyond the sysplex, the connection to the trackers on systems which are not in the sysplex will need to be either VTAM or shared DASD.

Note: All trackers do not have to be connected in the same way—for example, some might be using XCF to communicate with the controller, while other trackers in the OPCplex may be using VTAM connections.

There are a number of benefits of having just one OPCplex in the sysplex:

- It provides a single point of control for all OPC-controlled work in the sysplex.

  This means that an operator will only need to access one OPC controller to view all work in the sysplex.
- Jobs that run on one system can be dependent on jobs that run on another system.
- There is just one set of OPC databases and definitions to maintain.

How OPC submits jobs: If the system that an OPC-controlled job will run on is in the same MAS as the OPC controller, the controller will submit the job itself. This means that the controller needs access to the library containing the job JCL, and any PROCLIBs required for the job must be available to whichever system in the MAS might do the JCL conversion.

If the system that an OPC-controlled job will run on is *not* in the same MAS as the OPC controller, the controller will send the job JCL for each job to the OPC tracker on the target system, and the tracker will then submit the job. In this case, the controller needs access to the library containing the job JCL and the target system needs access to any PROCLIBs required by the job.

Except for the access to the job library and PROCLIBs just described, there is no need for anything else to be shared between the systems in the OPCplex.

If your target environment is a BronzePlex, the incoming system will not share any user DASD with the systems in the target sysplex, and it will also not be in the same MAS as any of those systems. Therefore, the OPC controller will send the job JCL to the tracker to submit. So, if you want to merge OPCs, there should not be a problem as long as you are able to store the JCL for the incoming system's OPC-controlled jobs on a volume that is accessible to the target sysplex systems (the incoming system does not need access to the JCL library, so the volume does not necessarily need to be shared between systems).

If you do decide *not* to merge the OPCs, there should be no reason you cannot move to the BronzePlex environment using two completely independent OPC subsystems. If you decide to go with this configuration, you will need to ensure that the XCF group and member names specified for OPC on the incoming system are different from the names used by OPC in the target sysplex.

If your target environment is a GoldPlex, the incoming system would again not be sharing user DASD with the other systems in the target sysplex, and would also normally not be in the same MAS as those systems. Once again, as long as you can make the incoming system's job's JCL accessible to the OPC controller, there should not be a problem if you wish to merge OPCs.

Finally, if your target environment is a PlatinumPlex, the incoming system *will* be sharing the user DASD with the other systems in the target sysplex. In this case, there is no impediment to merging OPC, at least from an accessibility point of view.

Regardless of which target environment you select, in most cases there is no obvious good technical reason *not* to merge the OPCplexes. The managability benefits are significant, and you do not lose any flexibility by having a single OPCplex.

17.2 Considerations for merging OPCplexes

In this section we provide a checklist of considerations for merging the OPCplexes. The major part of the merge process is the merging of the OPC databases. Not all of the OPC databases need to be merged. Generally speaking, only the databases which contain the definitions used to schedule the work need to be merged. OPC's two plan databases, the Long Term Plan (LTP) and the Current Plan (CP), do not need to be merged. Once the definitions databases have been merged, the LTP and CP can simply be regenerated using the normal OPC-provided planning batch jobs.

The degree of difficulty of the merge is determined by the number of entities with duplicate names that are contained in the databases of the two OPCplexes. In some cases the instances of duplicate names may be helpful, while in other cases it will require changes in the name or other details of the data.

Table 17-1 lists the items to be considered when merging OPCplexes. All these items are discussed in detail in 17.3, "Methodology for merging OPCplexes" on page 263.

Table 17-1 Considerations for merging OPCplexes

Consideration	Done
Check OPC databases for entities with duplicate names.	
Decide if the OPCplex will extend outside the sysplex.	
Determine if each system uses the same OPC optional functions, such as the Job Completion Checker (JCC), and address any disparities.	

Consideration	Done
Decide how the controller and trackers will be connected.	
Check the RACF definitions for OPC.	
Check OPC's own exits (EQQUXnnn).	
Check OPC-provided SMF exits.	
Check OPC-provided JES exits.	
WLM service class names for late jobs can be stored in OPC - these may need to be changed to cater for changed service class names in the merged WLM.	
Check if the OPC database data set sizes are sufficient.	
Investigate the performance of OPC to see if tuning is required to process the additional data.	

17.2.1 General notes about the merge project

There are some things we would like to point out about the OPC merge project.

We recommend that the actual OPC merge (that is, the point at which you move to having just one OPC controller) not be done until *after* the incoming system has been merged into the target sysplex. It should be possible to continue operating successfully with two OPCplexes in the sysplex until you are ready to merge the OPCs. We do *not* recommend merging the OPCs at the same time that you move the incoming system into the target sysplex—you want to keep the number of activities during that period as low as possible.

While it would be possible to merge OPCs before merging the incoming system into the target sysplex, you would need to make a change to use DASD or VTAM to connect the incoming system tracker to the controller, followed by another change after the system merge to change that connection to use XCF instead. Leaving the OPC merge until after the system merge reduces the number of changes you have to make to OPC.

Also, we have assumed that you have a test OPC subsystem that will be used solely for testing the changes you will make to the OPCs as part of the merge. The databases for the merge can be created by a DFSMSdss™ (or similar) copy of the databases from the OPCplex that you will be merging into—there is nothing within the OPC databases to indicate the OPC subsystem that owns those databases. You will also want to ensure that there is no way that the test OPC subsystem will try to submit any of the jobs that are defined to it. To do this, you should change the JTOPTS statement for the test OPC to, for example, JOBSUBMIT(NO), and to be doubly safe you should NOT move the production jobs into the EQQJBLIB until the actual day of the final merge. Then, even if the job submit function is activated via the dialogue, jobs that you do not want submitted will not be.

You should attempt to minimize the elapsed time for the merge. A certain amount of work will be needed to check each database and identify what changes will be required to complete the merge. Once this work starts, you should implement a change freeze on both OPCplexes; if you do not do this, incompatible changes could be made to a database after you have finished reviewing that database, resulting in problems when you subsequently try to merge the database.

When we start talking about merging the OPC databases, we have a generic process for most of the databases:

- Review the database in OPC in the incoming system and the target sysplex, looking for definitions with duplicate names.
- Unload the database definitions to a flat file using a sample exec that we provide.
- Manually edit the flat file to remove any duplicate definitions.

  In addition, if you want to change some of the definitions, you could make the change by editing the flat file using information about the record layouts that we provide in “Record Layout of REPORT1 File” on page 289. If you use this approach, the changes will not affect the OPC-managed work in the incoming system until you merge the OPC controllers.

  An alternate is to make any changes you need in OPC in the incoming system. The benefit of this approach is that you can see the impact of the change prior to the merge. An additional benefit is that it avoids having to edit the records in the flat file - an exercise that can be complex for records with complex record layouts.
- After you have made any required changes to the flat file, load the records into the target database using another sample exec we have provided. At this point, the target database should contain all the required definitions from both sets of databases.

17.3 Methodology for merging OPCplexes

In this section we discuss the approaches that can be taken to merging OPCplexes, and the tasks you will need to undertake to ensure a successful merge.

At the end of the merge, you will end up with one controller in the sysplex—presumably on the same system that the target sysplex OPCplex controller runs on today, and multiple trackers—one tracker for each system that runs OPC work. So, the incoming system, which starts with a controller and a tracker, ends up with just a tracker, connected via XCF to the controller running the OPCplex on one of the target sysplex systems. All the OPC database entries from the incoming system will have been merged into the OPC databases on the target OPCplex.

When the merge should occur

The bulk of the work involved in merging the OPCplexes can be done in advance of the day of the OPC merge, and this is our recommended approach. This is because a staged approach is generally safer than a “big bang” approach. This also allows plenty of time for checking the merged definitions, and making any corrections that may be required.

Because this work can be done in advance, the definitions that are to be merged can be cleaned up before they move into the target OPCplex and any changes to names of any of the OPC definitions can occur in advance.

The following OPC databases can be merged in advance:

- WS - Work Station
- CL - Calendars
- PR - Periods
- SR - Special Resources
- JCLVAR - JCL Variables

This is because the definitions in these databases are not used until the Application Description (AD) database has been merged. The data in these databases is only used by the definitions in the AD database.

The final merge is when the target OPCplex takes over scheduling and running the work that was previously controlled by OPC on the incoming system.

The final merge would consist of:

- Merging the Application Description (AD) database.
- Merging the Operator Instructions (OI) database.
- Merging the Event Triggered Tracking (ETT) database.
- Adding Special Resources from the Current Plan on the incoming system to the target OPCplex.
- Stopping and restarting the tracker on the incoming system using the new parameters which will connect this tracker to the target OPCplex controller via an XCF connection.
- Stopping and restarting the controller on the target OPCplex using the additional parameters to connect to the incoming system's tracker via an XCF connection.
- Running a Long Term Plan (LTP) Modify All batch job to update the LTP with the work from the incoming system.
- Extending the Current Plan (CP) to include the work from the incoming system.

We will look at the merge from two aspects:

1. Merging OPC system-related definitions. This consists of:
   - Merging the SMF and JES exits used by OPC.
   - Merging the RACF definitions for OPC and application data.
   - Merging OPC's own user exits.
   - Identifying a WLM service class to be used for late OPC jobs.
   - Agreeing subsystem names for the incoming tracker(s).
   - Investigating the use of OPC optional functions.
   - Checking on the use of OPC interfaces to Tivoli NetView®, SA for OS/390, and Tivoli InfoMan.
   - Creating initialization statements (for both the tracker and the controller) reflecting the new OPC environment.
   - Connecting the tracker on the incoming system to the existing OPCplex in the target sysplex.
   - Investigating OPC performance.
2. Merging the following OPC databases:
   - WS, CL, PR, SR, JCLVAR, AD, OI, and ETT.
   - Resizing the databases, where necessary.

As mentioned earlier, some of these databases can be merged in advance, while others would be done just prior to the final merge.

17.3.1 Merging OPC system-related definitions

In order to be able to monitor the status of jobs it submits, OPC uses a number of system exits. It also provides exits of its own, so you can alter various aspects of OPC's processing. There are also a number of parameters that you must specify to describe the operating environment to OPC. Some or all of these definitions may need to be changed as part of the merge; this section describes the considerations.

SMF and JES exits

OPC uses the following SMF exits to gather information about jobs and started tasks:

- SMF exit IEFUJI
- SMF exit IEFACTRT
- SMF exit IEFU84

If you are using JES2, then OPC also requires function in:

- JES2 EXIT7

If you are using JES3, then OPC requires function in:

- JES3 EXIT IATUX19
- JES3 EXIT IATUX29
- SMF exit IEFU83 (optional)

OPC supplies a standard version of each of these exits. It should not be necessary to modify that code. However, if you already use these exits for other functions, then the OPC function will need to be included. Regardless, the OPC-provided function should be identical on every system in the OPCplex. If you have not modified the OPC function as provided with OPC, you should not have to do anything with these exits, as the function should be identical on the incoming system and the target sysplex systems.

OPC user exits

Unlike the SMF and JES exits, OPC's own exits may differ from system to system, depending on why you want to use the exit on each system. The exits EQQUX001, EQQUX002, EQQUX003, EQQUX007, EQQUX009, and EQQUX011 are used by the controller. Exits EQQUX004, EQQUX005, EQQUX006, EQQUX008, EQQUX010 and EQQUX011 are used by the trackers. Exit EQQUX000 can be used by both the tracker and the controller. If these exits are being used, they need to be reviewed to determine:

- Are the exits still required?
- As the controller exits will be merged, will adding functions from the incoming system have an impact on the target sysplex OPCPlex?
- Can the exits be merged before the system merges into the OPCPlex?

The controller exits need to be merged (because a single controller will now be providing the function that was previously provided by two, independent, controllers) and therefore it is important to ensure the functions required by the OPCPlex and/or the incoming system are not lost. Although the tracker exits do not necessarily need to be merged, they should be reviewed to determine if the merging of the OPC data affected by these exits will cause any problems. For example, if the jobnames or application names change, any exit which refers to the old jobname or application name needs to be changed.

In addition, and especially in a PlatinumPlex, it is important that work is managed consistently, regardless of which system it runs on, so in that case you should merge the functions of any tracker exits so that the exit behaves in the same manner on all systems. The following is a list of the exits:

EQQUX000	OPC start/stop
EQQUX001	Job submit
EQQUX002	Job library read
EQQUX003	AD feedback
EQQUX007	Operation status change
EQQUX009	Operation initiation
EQQUX011	Job-tracking log write

EQQUX004	Event filtering
EQQUX005	JCC SYSOUT archiving
EQQUX006	JCC incident-record create
EQQUX008	Pre-catalog management
EQQUX010	Job-log retrieval
EQQUX011	Job-tracking log write

There are also a number of other exits which are called during various stages of OPC processing. These must also be considered before merging the OPCplexes. They are

User-defined	JCL imbed—called by the OPC FETCH directive found in JCL.
User-defined	JCL variable substitution.
User-defined	Automatic job recovery—called during JCL rebuild during automatic job recovery.
EQQDPUE1	Daily planning report—called by the daily planning batch job.

All of these exits are documented in *Tivoli Workload Scheduler for z/OS Customization and Tuning*, SH19-4544.

RACF definitions for OPC and application data

OPC uses RACF (or an equivalent security product) to protect its own databases and functions. The level of protection is decided by the installation. OPC can protect its data at the database level, or at the database record level. OPC uses fixed resources to protect at the database level, and subresources to protect at the database record level.

The subresources OPC uses to protect database records are based on one of the name fields in each of the databases. For example, you can limit access to OPC application descriptions using the application name. Therefore, if you need to change any of the names of the definitions or names used within the definitions and you use OPC subresources, you may need to change your RACF definitions. For a complete list of OPC fixed resources and OPC subresources, see *Tivoli OPC Customization and Tuning,* SH19-4380.

Because OPC will be submitting batch jobs, it needs to either have access to the data the jobs read and update, or have access to the userid used by the jobs. If there are new userids or data set names on the incoming system, then RACF will need to be changed to give OPC access.

If the RACF databases are being merged as part of the merge project, then the changes only need to be made in the one database. On the other hand, if the RACF databases remain separate, then changes may be required in both databases as follows:

- The database on the controller's system will need to include changes required to support the merged data. For example, new subresources to match the application names may need to be added.
- RACF in the incoming system may need to be changed if the userid associated with the tracker address space will be different than the userid that was previously used for the OPC-controlled jobs on that system prior to the merge.
- The catalog management actions of OPC are performed by the trackers. The trackers must therefore have access to all the production data sets that the batch jobs do.

WLM service class for late OPC jobs

OPC has the ability to assign a specific WLM service class to a critical job that is running late. The service class name is defined on the WLM keyword of the OPCOPTS statement in the OPC initialization parameters. If this feature is being used, the keyword may need to be changed if the service class names are different in the target sysplex (remember that there is only one WLM policy for the whole sysplex). For more information about this, refer to item 10 on page 61.

Subsystem names of incoming trackers

The subsystem name of the tracker on the incoming system can remain the same as long as it does not clash with the subsystem name of any of the existing trackers in the OPCplex. If, however, the tracker name must change to meet the naming standards of the OPCplex, then the following need to be changed:

- The subsystem name in the IEFSSNxx member of the incoming system.
- Any batch jobs or automation which issue any of the OPC TSO commands OPSTAT, SRSTAT, WSSTAT, OPINFO, BACKUP, JSUACT and use the tracker's name in the SUBSYS parameter.

OPC optional functions

There are a number of OPC optional functions which may be active on the incoming system and not be active on the OPCplex. Assuming that you still need these functions, they will need to be activated on the OPCplex. Changes will be required to the OPC initialization statements in order to activate these functions.

The optional functions and the associated initialization statements and parameters are as follows. Some parameters are used for more than one function:

- JCL Variable Substitution
  - OPCOPTS - VARSUB, VARFAIL, VARPROC
- Automatic Job Recovery (AJR)
  - OPCOPTS - ARPARM, RECOVERY
  - AROPTS - All parameters
- Catalog Management (CM)
  - OPCOPTS - CATMGT, CATACT, STORELOG
  - JOBOPTS - CATMCLAS, CATMDISP, CHSMWAIT, JOBLOGRETRIEVAL
- Job-log retrieval
  - OPCOPTS - JOBLOGDEST, STORELOG
  - JOBOPTS - CHSMWAIT, JOBLOGRETRIEVAL
- Event Triggered Tracking (ETT)
  - JTOPTS - ETT
- Job Completion Checker (JCC)
  - OPCOPTS - JCCTASK, JCCPARM
  - JCCOPTS - All parameters
- OPC Data Store
  - OPCOPTS - DSTTASK
  - DSTOPTS - All parameters
  - DSTUTIL - All parameters
  - FLOPTS - All parameters

- History Function
  - OPCOPTS - DB2SYSTEM
- Resource Object Data Manager (RODM)
  - OPCOPTS - RODMTASK, RODMPARM
  - RODMOPTS - All parameters

If these functions are already active in one or both of the OPCplexes, then the parameters will need to be reviewed during the merging of the initialization parameters. First, you should check if the same functions are being used in the target OPCplex. If not, then these initialization statements will need to be added to that system.

If the functions *are* in use by both OPCplexes, you will need to check the target OPCplexes initialization statements to ensure they are compatible with those from the incoming system. *Tivoli OPC Customization and Tuning,* SH19-4380, contains a description of all these parameters and can be used to determine how to code these initialization statements, or, if they are already in use, to determine the meaning of each statement.

OPC interfaces to Tivoli NetView, SA for OS/390 and Tivoli InfoMan

OPC uses a special workstation to interface to NetView. Part of this interface is a workstation defined in OPC with the prefix NV. The other part consists of code in OPC's exit 7. If you are currently using the OPC interface to Tivoli NetView, you will need to determine if the functions are still required and if so, make changes to OPC and NetView to ensure that NetView on the correct system receives the OPC request.

More information about this interface can be found in *System Automation for OS/390: OPC Automation Programmer's Reference and Operator's Guide*, SC33-7046. Further information about the InfoMan interface can be found in *Tivoli Information Management for z/OS: Integration Facility Guide Version 7.1*, SC31-8745.

Initialization statements

We have now addressed nearly all the OPC initialization statements that are likely to be affected by the merge. However, before proceeding, all the remaining OPC initialization statements in both environments should be compared to ensure there are no incompatible differences.

Connecting tracker on incoming system to the existing OPCplex

The final step (in terms of initialization statement-type changes) is to change the statements so that the tracker in the incoming system connects to the controller in the target sysplex. OPC trackers can be connected to the controller using any of three connection methods:

- XCF
- VTAM
- DASD

In a sysplex environment, we recommend using XCF to connect the trackers, so we will look at the changes required to the initialization parameters of the controller and the tracker to enable this.

Each tracker connects to an XCF group and identifies itself to XCF using a member name. The group name must be the same for all trackers and controllers in the OPCplex. The tracker member name must be unique in the OPCplex. The XCF groupname can be found in the initialization parameters of the controllers and trackers on the XCFOPTS statement under the GROUP parameter. The member name used for each tracker or controller is the name specified on the MEMBER parameter of the XCFOPTS statement when joining the XCF group. For example:

```
XCFOPTS GROUP(OPCPLEX) MEMBER(&SYSNAME.OPCC)
```

This statement could be used by the controller and standby controllers (the last character in the member name indicating that this is a controller).

The following statement could be used by all the trackers:

```
XCFOPTS GROUP(OPCPLEX) MEMBER(&SYSNAME.OPCT)
```

A good naming convention for the member name is to use the system name variable &SYSNAME (or the SMF ID, if the SYSNAME is longer than 4 characters) concatenated to the name of the started task. This structure allows you to share the OPC parmlib between all systems in the OPCplex.

These statements define each tracker and controller to XCF. Next, you need to define a path from the controller to each tracker, and from each tracker to the controller. A controller uses the XCF parameter of the ROUTEOPTS initialization statement to connect to each tracker. The name identified on the XCF parameter is the XCF member name that each tracker has specified as their member name on their XCFOPTS initialization statement. For example:

```
ROUTEOPTS XCF(SYS1OPCT,SYS2OPCT,SYS3OPCT)
```

This allows the controller to connect to three trackers; one on SYS1, one on SYS2, and one on SYS3. The above statement specifies that XCF is to be used to communicate from the controller to each of these three trackers.

Then you need to define a path from the tracker to the controller. This is done using the trackers HOSTCON parameter of the TRROPTS initialization statement. All you need to define here is that the connection to the controller is an XCF connection. For example:

```
TRROPTS HOSTCON(XCF)
```

This is all that is required to connect the trackers and controllers that reside in the same sysplex.

Important: Obviously, you would not activate these changes until the point at which you are ready to move to a single controller. However, the members could be set up in advance, and simply renamed at the time of the OPC controller merge.

Initialization changes to the controller and tracker for a VTAM connection

If you are currently using a VTAM connection between your controller and trackers that are in the same sysplex, we recommend that you take this opportunity to change those to an XCF connection.

Initialization changes to the controller and tracker for a DASD connection

If you are currently using a DASD connection between your controller and trackers that are in the same sysplex, we recommend that you take this opportunity to change those to an XCF connection.

OPC performance

OPC is capable of running over 100,000 jobs a day, so performance in most situations should not be an issue. If you are concerned about the impact on performance, review the IBM Redbook *Maximizing Your OPC/ESA Throughput,* SG24-2130, for tuning recommendations.

17.3.2 Merging the OPC databases

In this section, we look at each of the databases that need to be merged and how this may be achieved. Some of the OPC databases can be merged using OPC-supplied tools, while other databases will have to be merged using some other means—we have provided some sample programs to assist in those cases. In this section, we review the considerations for each of the databases, in the order that we suggest doing the merges. (In some cases, not all the data may need to be merged, as the existing definitions may be sufficient.)

Throughout this section we use the OPC Batch Command Interface Tool (BCIT) to unload a copy of the database definitions. We then use the same tool to reload some of the databases. BCIT is documented in *Tivoli Workload Scheduler for z/OS Programming Interfaces,* SH19-4545.

Where the BCIT does not support the unload/reload of a particular database, we discuss using the OPC Program Interface (PIF). There are four sample OPC PIF REXX execs in the Additional Materials for this redbook that can be used to unload and reload those databases not supported by the BCIT. All of the REXX execs are based upon the EQQPIFOP sample member provided in OPC's sample library SEQQSAMP. For more detailed information about the sample execs, see 17.4, "Tools and documentation" on page 286.

Note that the PIF does not support duplicate records. Specifically, if you try to use PIF to load an OPC definition, and the target database already contains a record with the same key, the load job will stop as soon as the duplicate record is hit, and it will end with a return code of 16. In this case, you must do something to remove the duplicate record:

- If the record really is a duplicate, and a matching definition already exists in the target database, then you can simply delete the duplicate record from the input file.
- If the record has a duplicate key, but the definitions are *not* identical, you may have to rename the resource in the incoming system so there are no longer two different definitions with the same name.

In either case, you will need to remove all the records that were already successfully processed from the input file—otherwise you will get a duplicate record as soon as you try to load the first record and the PIF exec will end with RC=16 again.

Note: OPC must be up and running in order to be able to run the PIF and BCIT execs.

Table 17-2 lists the OPC databases that must be considered during the merge, and indicates, for each one, what we will use to offload and reload the database definitions. It also indicates which OPC data set contains each database.

Table 17-2 OPC databases

Database	Unload exec	Reload exec	Merge necessary?	OPC data set
Workstations (WS)	UNLOAD	RELOAD	Yes	EQQWSDS
Calendar (CL)	UNLOAD	RELOAD	Yes	EQQWSDS

Database	Unload exec	Reload exec	Merge necessary?	OPC data set
Special Resources (SR)	UNLOAD and SRCPUNLD	RELOAD and SRSTAT job	Yes	EQQRDDS EQQCPxDS
Operator Instructions (OI)	Use BCIT	Use BATCHL	Yes	EQQOIDS
Current Plan (CP)	N/A	N/A	No	EQQCPxDS
Long Term Plan (LTP)	N/A	N/A	No	EQQLTDS
Period (PR)	UNLOAD	RELOAD	Yes	EQQWSDS
Event Triggered Tracking (ETT)	UNLOAD	RELOAD	Yes	EQQSIDS
JCL Variables (JCLVAR)	JCLCLREC followed by UNLOAD	RELOAD	Yes	EQQADDS
Application Description (AD)	Use BCIT	Use BATCHL	Yes	EQQADDS

We start by checking the definitions in the workstation, calendar, period, and special resources databases. The reason we address these databases first is because the definitions in these databases are referenced by entries in the application description database, and by the OPSTAT, SRSTAT, and WSSTAT TSO commands. If the names of any of the entries in the workstation, calendar, period, or special resource databases need to change, then any references to these names in the application description database or in batch jobs will also need to change.

As we will see, this should not cause too many problems as the process used to merge the application description database allows the use of TSO edit commands to change all references to each name with one command. Where batch jobs use OPSTAT, SRSTAT, or WSSTAT commands, edit macros can be used to make global changes to the affected jobs.

Tip: Where there are duplicate records and we suggest editing the file created by the UNLOAD exec, you should only ever change the NAME of the record. Do not try to edit any of the definitions in the unloaded file, as this would be prone to error. If you wish to change the definitions, do this using the OPC dialog before unloading the databases.

Workstation (WS) database

Tip: The workstation database can be merged in advance.

The workstation database is one of the databases where it is likely that not all of the definitions will need to be merged. For example, if the target sysplex OPCplex already has a computer automatic workstation called CPU1 and the incoming system OPCplex has an identical definition for the workstation called CPU1, and the systems will be in the same JES2 MAS, then there is no need to merge that definition.

However, if there are other workstations defined in the incoming system OPCplex, where an existing workstation in the target sysplex OPCplex will not suffice, you will need to add this workstation definition to the target sysplex OPCplex workstation database.

The first step is to look through the workstation definitions in both OPCplexes and determine if there are any duplicate workstation names. This can be done using the standard OPC ISPF front-end, or by using the Workstation Description Report that you can generate through the OPC panels, or through a printout that can be obtained using the Batch Command Interface Tool (BCIT).

Figure 17-1 on page 272 contains an extract from a sample Workstation Description Report.

```
-WORK STATION NAME   : CPU1                                  DEFAULTS
 DESCRIPTION         : Computer Automatic                     TRANSPORT TIME
                                                              DURATION
 WORK STATION TYPE   : COMPUTER
 REPORTING ATTRIBUTE : AUTO.
 PRINTOUT ROUTING    : SYSPRINT                               RESOURCE 1
 CONTROL ON SERVERS  : NO                                     NAME
                                                              PLANNING
 OPTIONS             :                                        CONTROL
  SPLITTABLE         : NO
  JOB SETUP          : NO                                     RESOURCE 2
  DESTINATION        :                                        NAME
  STARTED TASK       : NO                                     PLANNING
  WTO                : NO                                     CONTROL
- LAST UPDATED BY    : U104208    ON 03/10/02   AT 16.46
                       DAY/DATE      OPEN TIME          PARALLEL   RESOURCES  A
```

Figure 17-1 Sample Workstation report generated from OPC ISPF panels

Figure 17-2 contains an extract from a sample workstation report from BCIT.

```
WORKSTATION
                                 _____________________
WORKSTATION ID                  :CPU1
DESCRIPTION                     :Computer Automatic
INTERVALS NUMBER                :       1
WORKSTATION TYPE                :C
REPORTING ATTRIBUTE             :A
CONTROL ON PARALLEL SERVERS     :N
R1 NAME                         :CA
R1 USED AT CONTROL              :N
R2 NAME                         :TA
R2 USED AT CONTROL              :N
SETUP ABILITY                   :N
RECORD VERSION NUMBER           :01
STARTED TASK OPTION             :N
WTO MESSAGE OPTION              :N
DEFAULT DURATION in SEC*100     :       0
FAULT-TOLERANT WS               :N
```

Figure 17-2 Sample workstation report from BCIT

If there are any duplicate workstation names, you need to determine if the workstation characteristics are the same. If there are any workstations with the same name, but different characteristics, then the names of those workstations will need to be changed before merging them into the target sysplex OPCplex database (unless, of course, you can bring the definitions into line without impacting any of the work).

You can change the workstation name by editing the output file produced from the sample OPC PIF exec called UNLOAD before reloading the data with the sample OPC PIF exec called RELOAD.

Note: Keep track of any workstations you have renamed, and the new names that you used—you will need this information later on in the merge.

If all the workstations are defined the same, there is no need to reload the definition—just use the existing workstation definitions.

If there are workstations that are only defined in the incoming system OPCplex, then those will need to be added to the target sysplex OPCplex. To do a selective merge, the output file from the UNLOAD sample exec must be edited to remove any workstations which do not need to be merged before running the RELOAD exec.

Using the provided sample execs to merge the WS database

The WS database has variable length records (LRECL); the maximum LRECL of a WS record is 32000 bytes. Therefore, the file used as the output file by the UNLOAD exec, and as the input file by the RELOAD exec, has an LRECL of 32000. The UNLOAD exec determines the LRECL of each database record as it runs and writes records of the correct length to the WS file. There are four steps to merge the WS database:

1. Allocate the WS output file

This file can either be allocated in the batch job which runs the UNLOAD exec, or in advance using ISPF. The LRECL is 32000 and the RECFM is FB. It is placed on the WS DD statement in the BATCH2 job (shown in Figure 17-3). See “Program Interface samples” on page 287 for more information about the BATCH2 job.

```
//IBMUSERA JOB (ISC),'OPC PIF',CLASS=A,MSGCLASS=X,MSGLEVEL=(1,1),NOTIFY=&SYSUID
//*--------------------------------------------------------------------
//* THE PARM CARD ON THE EXEC CARD IS EITHER
//* UNLOAD - UNLOADS THE DATABASE IDENTIFIED ON SYSTSIN
//* RELOAD - RELOADS THE DATABASE IDENTIFIED ON SYSTSIN
//*
//* DDNAME         PURPOSE
//* --------       -----------------
//* SYSEXEC        DATA SET CONTAINING REXX EXECS
//* EQQMLIB        OPC MESSAGE DATA SET (SEQQMSG0)
//* CL,ETT,JCLV,   OUTPUT DATA SETS FOR UNLOAD EXEC OR INPUT DATASET
//* PR,SR,WS       FOR RELOAD EXEC - ALL HAVE LRECL=32000 AND RECFM=FB
//*--------------------------------------------------------------------
//STEP1   EXEC PGM=IKJEFT01,DYNAMNBR=30,REGION=4096K,
//             PARM='UNLOAD'
//SYSEXEC   DD DISP=SHR,DSN=U104208.OPCMERGE.REXX
//EQQMLOG   DD SYSOUT=*
//EQQMLIB   DD DISP=SHR,DSN=SYS1.TWSZOS.SEQQMSG0
//SYSTSPRT  DD SYSOUT=*
//CL       DD DISP=SHR,DSN=U104208.CL.REPORT1
//ETT      DD DISP=SHR,DSN=U104208.ETT.REPORT1
//JCLV     DD DISP=SHR,DSN=U104208.JCLV.REPORT1
//PR       DD DISP=SHR,DSN=U104208.PR.REPORT1
//SR       DD DISP=SHR,DSN=U104208.SR.REPORT1
//WS       DD DISP=SHR,DSN=U104208.WS.REPORT1
//SYSTSIN  DD *
 specify the OPC subsystem name followed by CL, ETT, JCLV, PR, SR, or WS as in the
following example.....
TWSC WS RO*
/*
//
```

Figure 17-3 Sample BATCH2 JCL

2. Unload the WS database using the UNLOAD exec.

   Use the sample job BATCH2 to run the UNLOAD exec. Change the SYSEXEC and EQQMLIB DD statements to point to your data sets. The UNLOAD exec is identified on the PARM on the EXEC card as PARM='UNLOAD'. The SYSTSIN DD * statement should contain "xxxx WS nnn*", where xxxx is the name of the incoming system's OPC subsystem, and nnn* is the name of the workstation that you want to unload (see "Program Interface samples" on page 287 for more information about this parameter). The UNLOAD exec unloads all the WS database records to the WS output file. Each record is written to one line of the file. This exec should finish with a zero return code.

3. Edit the WS output file to make any required changes.

   Using the ISPF editor, change the names of any workstations that must be renamed and delete all duplicate records and any other records which you do not want to load in to the target OPCplex. See 17.4, "Tools and documentation" on page 286 for an explanation of the record layout of each database record in the WS output file.

4. Load the WS database definitions into the target OPCplex using the RELOAD exec.

   Use the sample job BATCH2 to run the RELOAD exec. Change the SYSEXEC and EQQMLIB DD statements to point to your data sets. The RELOAD exec is identified on the PARM on the EXEC card as PARM='RELOAD'. The SYSTSIN DD * statement should contain "yyyy WS", where yyyy is the name of the target sysplex OPC subsystem.

The RELOAD exec loads all the WS database records from the WS file into the target OPCplex. This exec should finish with a zero return code. If it ends with a return code of 16, the problem is probably that the input file contains a duplicate record. In this case, you should see a message similar to the following:

```
EQQY724E INSERT OF RECORD WITH KEY xxxxxx
EQQY724I NOT DONE, EXISTS. RESOURCE IS WS.
```

The text after the word KEY in the first message identifies the duplicate record.

This completes the merge process for the WS database.

Calendar (CL) database

Tip: The calendar database can be merged in advance.

The calendar database describes the working week to OPC—that is, which days are normally considered days on which work will run, and which days work does not normally run. Days on which work runs are described as “work” days, and days on which work does not normally run are described as “free” days. If work does not normally run on public holidays, these dates are defined as “free” days. The calendar database is another database where not all of the definitions may need to be merged. If both OPCplexes calendars describe the working week in the same manner, there is no need to merge the definitions.

The first step is to look through the calendar definitions in both OPCplexes and determine if there are any duplicate calendar names. This can be done using the standard OPC ISPF front-end, or by using the Calendar Reports that you can generate through the OPC panels, or through a printout that can be obtained using BCIT.

The merge process is similar to the process we used for the workstation database. That is, use the sample UNLOAD exec to unload the database, make any changes that may be required to the unloaded file (deleting duplicate records, for example), and then use the sample RELOAD exec to load the information into the database of the target sysplex OPC.

If there are any calendars with the same name but different characteristics, the names will need to be changed by editing the output from the unload exec before using the RELOAD exec. You can do this by editing the output file produced by the UNLOAD exec.

Note: Keep track of any calendars you have renamed, and the new names that you used—you will need this information later on in the merge.

If the calendars are identical, then there is no need to rename or redefine them—just continue using the calendars already defined in the target sysplex OPCplex calendar database.

However, if there are calendars in the incoming system OPCplex that are not defined in the target sysplex OPCplex, you must add those definitions. If you have calendar entries which do not need to be merged, they need to be deleted from the ouptut file produced from the UNLOAD exec before running the RELOAD exec.

Using the provided sample execs to merge the CL database

The CL database has variable length records (LRECL); the maximum LRECL of a CL record is 32000 bytes. Therefore, the CL file which is used as an output file by the UNLOAD exec and an input file by the RELOAD exec has an LRECL of 32000.

The UNLOAD exec determines the LRECL of each database record as it runs, and writes records of the correct length to the CL file. There are four steps to merge the CL database:

1. Allocate the CL output file.

   This file can either be allocated in the batch job which runs the UNLOAD exec, or by you using ISPF. The LRECL is 32000 and the RECFM is FB. It is placed on the CL DD statement in the BATCH2 job.

2. Unload the CL database using the UNLOAD exec.

   Use the sample job BATCH2 to run the UNLOAD exec. Change the SYSEXEC and EQQMLIB DD statements to point to your data sets. The UNLOAD exec is identified on the PARM on the EXEC card as PARM='UNLOAD'. The SYSTSIN DD * statement should contain "xxxx CL nnn*", where xxxx is the name of the incoming system's OPC subsystem, and nnn* is the name of the workstation that you want to unload (see "Program Interface samples" on page 287 for more information about this parameter).

   The UNLOAD exec unloads all the CL database records to the CL output file. Each record is written to one line of the file. This exec should finish with a zero return code.

3. Edit the CL output file, as required.

   Change the names of any calendars as required and delete all duplicate records and any records which you do not want to load in to the target OPCplex. See 17.4, "Tools and documentation" on page 286 for an explanation of the record layout of each database record in the CL output file.

4. Load the CL database definitions into the target OPCplex using the RELOAD exec.

   Use the sample job BATCH2 to run the RELOAD exec. Change the SYSEXEC and EQQMLIB DD statements to point to your data sets. The RELOAD exec is identified on the PARM on the EXEC card as PARM='RELOAD'. The SYSTSIN DD * statement should contain "yyyy CL", where yyyy is the name of the target sysplex OPC subsystem.

   The RELOAD exec loads the CL database records to the target OPCplex. This exec should finish with a zero return code. If it ends with a return code of 16, the problem is probably that the input file contains a duplicate record. In this case, you should see a message similar to:

   ```
   EQQY724E INSERT OF RECORD WITH KEY xxxxxx
   EQQY724I NOT DONE, EXISTS. RESOURCE IS CL.
   ```

   The text after the word KEY in the first message identifies the duplicate record.

This completes the merge process for the CL database.

Period (PR) database

Tip: This database can be merged in advance.

The period database is used by OPC when determining when to run jobs. The period database is another database where not all of the definitions need to be merged. The merge process is similar to the workstation database.

First, you must identify if there are duplicate periods. To do this, view the period definitions for both OPCplexes using the OPC ISPF panels, or print the period definitions using the report option in the OPC panels (the program to print the periods is EQQCLPRP). If there are any periods with the same name but different characteristics, the names will need to be changed or (preferably) the definitions brought into line if possible. You can do this by making the changes though the OPC ISPF panels, or by editing the output file from the UNLOAD exec.

Note: Keep track of any calendars you have renamed, and the new names that you used—you will need this information later on in the merge.

If the period definitions are the same, there is no need to rename or redefine them; just continue using the periods already defined in the target sysplex OPCplex period database. However, if there are periods in the incoming system OPCplex that are not defined in the target sysplex OPCplex, you must add those definitions.

Once you have identified what changes need to be made, and which definitions need to be merged, you are ready to proceed. Once again, the merge process is similar to the WS database—namely, unload the database using the sample UNLOAD exec, make any changes that may be required in the resulting flat file, and then run the RELOAD exec to load the definitions into the target sysplex OPC PR database.

Using the provided sample execs to merge the PR database

The PR database has variable length records (LRECL). However, the maximum LRECL of a PR record is 32000 bytes. Therefore, the PR file that is used as an output file by the UNLOAD exec and an input file by the RELOAD exec has an LRECL of 32000.

The UNLOAD exec determines the LRECL of each database record as it runs and writes records of the correct length to the PR file. There are four steps to merge the PR database as follows:

1. Allocate the PR output file.

   This file can either be allocated in the batch job which runs the UNLOAD exec, or by you using ISPF. The LRECL is 32000 and the RECFM is FB. It is placed on the PR DD in the BATCH2 job.

2. Unload the PR database using the UNLOAD exec.

   Use the sample job BATCH2 to run the UNLOAD exec. Change the SYSEXEC and EQQMLIB DD statements to point to your data sets. The UNLOAD exec is identified on the PARM on the EXEC card as PARM='UNLOAD'. The SYSTSIN DD * statement should contain "xxxx PR nnn*", where xxxx is the name of the incoming system's OPC subsystem, and nnn* is the name of the workstation that you want to unload (see "Program Interface samples" on page 287 for more information about this parameter).

   The UNLOAD exec unloads all the PR database records to the PR output file. Each record is written to one line of the file. This exec should finish with a zero return code.

3. Edit the PR output file as required.

   Change the names of any periods as required, and delete any duplicate records or any other records that you do not want to load to the target OPCplex. See 17.4, "Tools and documentation" on page 286 for an explanation of the record layout of each database record in the PR output file.

4. Load the PR database definitions into the target OPCplex using the RELOAD exec.

   Use the sample job BATCH2 to run the RELOAD exec. Change the SYSEXEC and EQQMLIB DD statements to point to your data sets. The RELOAD exec is identified on the PARM on the EXEC card as PARM='RELOAD'. The SYSTSIN DD * statement should contain "yyyy PR", where yyyy is the name of the target sysplex OPC subsystem.

   The RELOAD exec loads the PR database records to the target OPCplex. This exec should finish with a zero return code. If it ends with a return code of 16, the problem is probably that the input file contains a duplicate record. In this case, you should see a message similar to:

   ```
   EQQY724E INSERT OF RECORD WITH KEY xxxxxx
   EQQY724I NOT DONE, EXISTS. RESOURCE IS PR.
   ```

The text after the word KEY in the first message identifies the duplicate record.

This completes the merge process for the PR database.

Special Resource (SR) database

Note: Some Special Resources can be merged in advance of the merge of the OPC subsystems. This section must be used in conjunction with “Special Resources in the Current Plan” on page 279.

Special resources are typically defined to represent physical or logical objects used by jobs. For example, a special resource can be used to serialize access to a data set, or to limit the number of file transfers on a particular network link. The resource does not have to represent a physical object in your configuration, although it often does.

When preparing to merge the SR databases, you need to check if there are any SRs with the same name but different characteristics, for example if there are different numbers of the resource defined in one database than in the other. If this is the case, you will either need to change the name of the special resource, or bring the two definitions into line (which might not be easy or even possible prior to the merge of the OPC subsystems).You can easily change the name of the resource by editing the output file from the UNLOAD exec.

Note: If you need to change the name of any SRs, you must keep a record of the old and new names—you will need this information later on.

Tip: An SR can be used to represent physical system resources such as cartridge or tape drives. If you use this type of SR, it is likely that the SR name used to represent the drives on the incoming system is the same as the name used on the target system. In this case, it may not be possible to merge these SRs until the point at which the OPC subsystems are being merged.

For example, if you have an SR that represents the number of tape drives available to the OPCplex, and there are 10 drives in the target sysplex and 5 in the incoming system, you cannot change that definition to, for example, 15 until you merge the OPCs.

If you have this type of SR, you should delete them from the flat file produced by the UNLOAD exec, and make a note that you need to go back and address them manually at the time of the merge.

If all the SRs are identical (which is unlikely), there is no need to rename or redefine them, or even to run the merge job—you can just use your existing SRs in the target sysplex OPCplex.

Using the provided sample execs to merge the SR database

The SR database has variable length records (LRECL). However, the maximum LRECL of a SR record is 32000 bytes. Therefore, the SR file which is used as an output file by the UNLOAD exec and an input file by the RELOAD exec has an LRECL of 32000.

The UNLOAD exec determines the LRECL of each database record as it runs and writes records of the correct length to the SR file. There are four steps to merge the SR databases, as follows:

1. Allocate the SR output file.

   This file can either be allocated in the batch job which runs the UNLOAD exec, or by you using ISPF. The LRECL is 32000 and the RECFM is FB. It is placed on the DD statement with a DDNAME of SR in the BATCH2 job.

2. Unload the SR database using the UNLOAD exec.

   Use the sample job BATCH2 to run the UNLOAD exec. Change the SYSEXEC and EQQMLIB DD statements to point to your data sets. The UNLOAD exec is identified on the PARM on the EXEC card as PARM='UNLOAD'. The SYSTSIN DD * statement should contain "xxxx SR nnn*", where xxxx is the name of the incoming system's OPC subsystem, and nnn* is the name of the workstation that you want to unload (see "Program Interface samples" on page 287 for more information about this parameter).

   The UNLOAD exec unloads all the SR database records to the SR output file. Each record is written to one line of the file. This exec should finish with a zero return code.

3. Edit the SR output file as required.

   Change the names of any special resources, or delete any records which you do not want to load to the target OPCplex. See 17.4, "Tools and documentation" on page 286 for an explanation of the record layout of each database record in the SR output file.

4. Load the SR database definitions into the target OPCplex using the RELOAD exec.

   Use the sample job BATCH2 to run the RELOAD exec. Change the SYSEXEC and EQQMLIB DD statements to point to your data sets. The RELOAD exec is identified on the PARM on the EXEC card as PARM='RELOAD'. The SYSTSIN DD * statement should contain "yyyy SR", where yyyy is the name of the target sysplex OPC subsystem.

   The RELOAD exec loads all the SR database records in the input file to the target OPCplex. This exec should finish with a zero return code. If it ends with a return code of 16, the problem is probably that the input file contains a duplicate record. In this case, you should see a message similar to:

   ```
   EQQY724E INSERT OF RECORD WITH KEY DEFAULT
   EQQY724I NOT DONE, EXISTS. RESOURCE IS SR.
   ```

   The text after the word KEY in the first message identifies the duplicate record.

Special Resources in the Current Plan

Tip: Special Resources in the Current Plan (CP) should be merged at the time of the merge of the OPC subsystems, and not in advance.

SRs are unique in that they can exist in two OPC databases at the same time. The defaults for all SRs are held in the Resource Database (RD) database, where we refer to them as "SRs". When the CP is extended, any SRs referenced by jobs coming into the CP will cause a copy of the SR to be placed in the CP database, where it is called a "CSR". Each time the CP is extended, a new copy of the referenced SRs is copied into the CP with the default values.

Once a CSR has been generated in the CP, the operator can override the default values using the Modify Current Plan (MCP) option of the OPC ISPF dialog, or by using the SRSTAT command or the Program Interface (PIF). The next time the CP is extended, any of the default values changed by the operator, SRSTAT, or PIF are reset to the defaults found in the RD database, with three exceptions:

- Availability
- Quantity
- Deviation

If any of these attributes have been altered, the altered values remain in force indefinitely. When the CP is next extended, these values are *not* returned to the default values which are specified in the RD database. To nullify these overrides, requires use of the RESET keyword against each attribute.

While CSRs can be unloaded with the UNLOAD exec, you cannot use the RELOAD exec to load the definitions back into OPC. Therefore, a special exec called SRCPUNLD has been written to handle these resources—you should *not* use the UNLOAD/RELOAD execs for these resources.

The SRCPUNLD exec produces a batch job which can be run to create CSRs in the CP of the target OPCplex. This job uses the SRSTAT command. The SRCPUNLD exec takes into account any of the three attributes that have been overridden, and generates an SRSTAT command, which will cause these attributes to show up as overridden in the target OPCplex. For example, if the CSR has no values overridden, the SRSTAT command would look like:

```
SRSTAT 'SPECIAL_RESOURCE1' SUBSYS(MSTR)
```

However, if the CSR has the availability attribute overridden, the command would look like:

```
SRSTAT 'SPECIAL_RESOURCE1' SUBSYS(MSTR) AVAIL(Y)
```

Each CSR that has one of the three attributes overridden will have the name of that attribute added to the SRSTAT command with the overridden value. If there is a need to change the name of any of the CSRs before merging them with the target OPCplex, the names can be altered in the generated batch job before running it.

Using the SRCPUNLD exec to merge the CSRs into the CP

The SRCPUNLD does not unload the SRs in the CP like the UNLOAD exec. It generates a batch job which must then be run to reload the SRs into the CP of the target OPCplex. The reload is performed by issuing SRSTAT commands for each CSR. Therefore, the LRECL of the file used to contain the generated batch job should be 80. There are four steps to merge the SR databases, as follows:

1. Allocate the CSR output file.

   This file can either be allocated in the batch job which runs the SRCPUNLD exec, or by you, using ISPF. The LRECL is 80 and the RECFM is FB. It is placed on the DD statement with a DDNAME of SR in the BATCH2 job.

2. Create the batch job by running the SRCPUNLD exec.

   Use the sample job BATCH2 to run the SRCPUNLD exec. Change the SYSEXEC and EQQMLIB DD statements to point to your data sets. The SRCPUNLD exec is identified on the PARM on the EXEC card as PARM='SRCPUNLD'. Place "xxxx CSR" on the SYSTSIN DD * statement, where xxxx identifies the OPC subsystem on the incoming system.

   The SRCPUNLD exec reads all the CSR database records and creates the batch job used to define the CSRs in the target OPCplex. This exec should finish with a zero return code.

3. Edit the CSR output file as required.

   Change the names of any CSRs that must be renamed, or delete any records which you do not want to load into the target OPCplex.

4. Load the CSR database definitions into the target OPCplex by running the generated batch job.

   The batch job that runs all the SRSTAT commands is built in the data set identified on the CSR DD statement in the BATCH2 job. Submit this job to define the CSRs in the target OPCplex. This job should finish with a return code of 0.

   Unlike the RELOAD exec used for the other databases, this job will never fail because of duplicate records. If a CSR already exists in the CP, the command updates the existing CSR with any changed values for availability, quantity and deviation.

This completes the merge process for the SRs in the CP.

JCL Variable Tables (JCLVAR)

The merge process for the JCLVAR database is also similar to that used with the workstation database. That is, you would normally use the sample UNLOAD exec to unload the database, make any required changes in the unloaded file, and then use the sample RELOAD exec to load the required database definitions to the target OPCplex.

There is one difference for the JCLVAR database—whereas the other databases have a maximum record length of 32000, it is possible for the JCLVAR database to contain records larger than this. Therefore, an additional exec called JCLCLREC is provided to identify all JCLVAR records that have a record length (LRECL) larger than 32000.

If you find that you do have such records, you must manually delete them from the output file produced by the UNLOAD exec. The records that you delete will need to be merged manually, using the OPC ISPF dialog.

As an alternative, if the JCLCLREC exec identifies records with an LRECL greater than 32000, the UNLOAD exec could be altered to unload only records with an LRECL equal to or less than 32000, and identify (but skip over) all records with a LRECL greater than 32000.

Note: Remember that if you alter the exec to skip some records, those records must be handled manually.

If there are any JCLVAR tables with the same name but different characteristics, the table names will need to be changed. You can do this by editing the output file from the UNLOAD exec. If you need to change the name of any JCLVAR tables, you must keep a record of the old and new names as you will need this information later on.

If the JCLVARs are the same, then there is no need to rename and redefine them; just use your existing JCLVAR tables.

Using the provided sample execs to merge the JCLVAR database

The JCLVAR database has variable length records (LRECL), with a maximum LRECL of 131072 bytes. However, you can only use ISPF to edit a file with a maximum LRECL of 32760. For consistency with all the other output files, the LRECL of the JCLV file, which is used as an output file by the UNLOAD exec and an input file by the RELOAD exec, has an LRECL of 32000.

The UNLOAD exec determines the LRECL of each database record as it runs, and writes records of the correct length to the JCLV file.

There are five steps to merge the JCLVAR database:

1. Determine if any JCLVAR records have an LRECL larger than 32000 by running the JCLCLREC exec.

   The sample BATCH1 job can be used to run this exec in batch mode. Change the SYSEXEC and EQQMLIB DD statements to point to your data sets. The JCLCLREC exec is identified on the SYSTSIN DD * statement. The following message identifies JCLVAR records which need to be merged manually and is written to sysout:

   ```
   The logical record length of JCL Variable Table VARDEPS
   is larger than 32000 bytes.
   Merge this table manually using the OPC ISPF dialogs.
   ```

   VARDEPS is the name of the variable table, in this example. If there are multiple records longer than 32000 bytes, all of those records will be listed.

   If you have any JCL variable tables which have an LRECL greater than 32000, those records will be truncated to 32000 bytes in the output file produced by the UNLOAD exec. In this case you have two choices:

   a. Perform the merge manually using the OPC ISPF dialog. This could be time-consuming.

   b. Identify the long records and manually delete them from the output file. The RELOAD exec can then be used to load the remaining records into the target OPCplex.

   This exec should finish with a 0 return code.

2. Allocate the JCLV output file.

   This file can either be allocated in the batch job which runs the unload exec, or by you, using ISPF. The LRECL is 32000 and the RECFM is FB.

3. Unload the JCLVAR database using the UNLOAD exec.

   Use the sample job BATCH2 to run the UNLOAD exec. Change the DD statements of SYSEXEC and EQQMLIB to point to your data sets. The UNLOAD exec is identified on the PARM on the EXEC card as PARM='UNLOAD'. The SYSTSIN DD * statement should contain "xxxx JCLV nnn*", where xxxx is the name of the incoming system OPC subsystem, and nnn* is the name of the workstation that you want to unload (see "Program Interface samples" on page 287 for more information about this parameter).

   The UNLOAD exec unloads all the JCLVAR database records to the JCLV file. Each record is written to one line of the file. This exec should finish with a zero return code.

4. Edit the JCLV file as required.

   Change the names of any JCL variable tables, or delete any records which you do not want to reload in to the target OPCplex. See 17.4, "Tools and documentation" on page 286 for an explanation of the record layout of each database record in the REPORT1 file.

5. Reload the JCLV database into the target OPCplex using the RELOAD exec.

   Use the sample job BATCH2 to run the RELOAD exec. Change the DD statements of SYSEXEC and EQQMLIB to point to your data sets. The RELOAD exec is identified on the PARM on the EXEC card as PARM='RELOAD'. The SYSTSIN DD * statement should contain "yyyy JCLV", where yyyy is the name of the target sysplex OPC subsystem.

   The RELOAD exec reloads all the JCLVAR database records to the target OPCplex. This exec should finish with a zero return code.

This completes the merge process of the JCLV database.

The Application Description (AD) database

Tip: The unload of this database and any editing of the data to be loaded can be done in advance. However, the actual loading of the data into the target sysplex OPC should *not* be done until during the final merge.

Obviously, if you take this approach you must ensure that there are no changes to the AD definitions subsequent to running the unload.

The AD database is a database whose entire contents need to be merged. Once again, if there are already applications defined in the target sysplex OPCplex with the same name as the applications to be merged, the names of the affected applications in the incoming system OPCplex will need to be changed.

However, simply checking for duplicate application names is not sufficient. Each application contains jobs. If any of the jobs from the incoming system have the same names as jobs in OPC in the OPC target system, those job names will need to be changed. You can check for this by using SUPERC to compare the member names in the OPC job libraries from both OPCplexes - any duplicates will need to be addressed.

Also, if any of the workstations, calendars, periods, Special Resources, or JCL variable tables names have changed, the references to these names in the ADs will need to change. These references can be changed by editing the output file produced by the BCIT when the AD database is unloaded from the incoming system OPCplex.

The following list shows you where each of the other databases may be referenced in an application description.

Workstation

Workstations can be referenced in an AD in:

1. The operations.

   Each operation is defined in the AD using a WS name.

2. External dependencies and extra internal dependencies.

   External dependencies refer to the AD name, WS name, Operation number, and Jobname. Up to eight internal dependencies can be defined on the operations panel. If they are defined on this panel, there is no need to specify a WS name. You only need to use an operation number.

   If there are more than eight internal dependencies for an operation, you will need to select the operation and then choose the dependencies option. This is where you defined external dependencies and internal dependencies if you have more than eight. In the dependency panel, you will need to use a WS name, as well as an operation number, to define extra internal dependencies.

Calendar

Calendars are referenced on the AD general information panel.

Periods

Periods are referenced in the AD run cycle.

JCL variable tables

JCL variable table names can be referenced in a number of places:

- The calendar
- The period

- The AD run cycle
- Within the JCL itself

Special Resources

Special Resources can be referenced in the operation descriptions within each application.

Once you know which definitions have been renamed in the associated databases, then reflecting those name changes in the AD database should not be difficult. We have provided a sample job called BCIT which uses the BCIT to unload the AD database data from the merging OPCplex.

The following BCIT command can be used to unload the data:

ACTION=LIST,RESOURCE=ADCOM,ADID=*.

The ADID parameter can be specific, or you can use wild card characters. So, if you wish to unload all applications with the prefix FRED, for example, ADID can be specified as:

ACTION=LIST,RESOURCE=ADCOM,ADID=FRED*.

This command produces a file containing batch loader statements. After you have made any required changes to the statements (to reflect any definitions that were renamed), this file can then be used to load the data to the target OPCplex. For example, if a workstation name has changed from MAN1 to MNL1, you can use the ISPF **CHANGE** command to change all the references in the batch loader statements file, as follows:

```
C MAN1 MNL1 ALL
```

Once the output file has been edited and changed as required, it can be used as input to the batch loader program. We have provided another sample called BATCHL can be used to execute the batch loader. Details of the BCIT and batch loader are provided in 17.4, “Tools and documentation” on page 286.

Operator Instruction (OI) database

Tip: If you have changed the names of any applications or jobnames in the target system during the merge, remember that any OI definitions related to these entities also need to be changed.

The OI database can be unloaded and reloaded using the same process as the AD database. The following BCIT command can be used to unload the data:

ACTION=LIST,RESOURCE=OICOM,ADID=*.

This command produces a file containing batch loader statements. This file is then used as input to the batch loader program. You should make any changes as required based on workstations, applications, or operations that you may have renamed. Once you have updated the batch loader statement, you can use the provided sample BATCHL job to invoke the batch loader to load the definitions into the target sysplex OPC OI database.

Event Triggered Tracking (ETT) database

The ETT database is unique in that all the definitions are shown on a single OPC ISPF panel. Therefore, it is possible to simply cut from the MODIFY ETT CRITERIA panel on the incoming system and paste the information to the MODIFY ETT CRITERIA panel on the target OPCplex. However, this may not be suitable if you have hundreds of definitions, in which case you should use the sample UNLOAD to unload the database.

If there are any ETT criteria with the same name but different characteristics, then the names will need to be changed. You can do this by editing the output file from the UNLOAD exec. If the ETT criteria are the same, then there is no need to rename and redefine them. Just use your existing ETT criteria. The information can be loaded into the target sysplex OPC ETT database by using the RELOAD exec.

Using the provided sample execs to merge the ETT database

The ETT database has fixed logical length records (LRECL). The LRECL of an ETT record is 200 bytes. However, for consistency, the file that is used as an output file by the UNLOAD exec and an input file by the RELOAD exec has an LRECL of 32000. There are four steps to merge the ETT database, as follows:

1. Allocate the ETT output file.

   This file can either be allocated in the batch job which runs the UNLOAD exec, or by you, using ISPF. The LRECL is 32000 and the RECFM is FB. It is placed on the DD statement with a DDNAME of ETT in the BATCH2 job.

2. Unload the ETT database using the UNLOAD exec.

   Use the sample job BATCH2 to run the UNLOAD exec. Change the DD statements of SYSEXEC and EQQMLIB to point to your data sets. The UNLOAD exec is identified on the PARM on the EXEC card as PARM='UNLOAD'. The SYSTSIN DD * statement should contain "xxxx ETT nnn*", where xxxx is the name of the incoming system's OPC subsystem, and nnn* is the name of the workstation that you want to unload (see "Program Interface samples" on page 287 for more information about this parameter).

   The UNLOAD exec unloads all the ETT database records to the ETT output file. Each record is written to one line of the file. This exec should finish with a zero return code.

3. Edit the ETT output file as required.

   Change the names of any ETT criteria or delete any records which you do not want to load to the target OPCplex. See 17.4, "Tools and documentation" on page 286 for an explanation of the record layout of each database record in the ETT output file.

4. Load the ETT database definitions into the target OPCplex using the RELOAD exec.

   Use the sample job BATCH2 to run the RELOAD exec. Change the DD statements of SYSEXEC and EQQMLIB to point to your data sets. The RELOAD exec is identified on the PARM on the EXEC card as PARM='RELOAD'. The SYSTSIN DD * statement should contain "yyyy ETT", where yyyy is the name of the target sysplex OPC subsystem.

   The RELOAD exec loads the ETT database records to the target OPCplex. This exec should finish with a zero return code. If it ends with a return code of 16, the problem is probably that the input file contains a duplicate record. In this case, you should see a message similar to:

   ```
   EQQY724E INSERT OF RECORD WITH KEY DEFAULT
   EQQY724I NOT DONE, EXISTS. RESOURCE IS ETT.
   ```

   The text after the word KEY in the first message identifies the duplicate record.

This completes the merge process for the ETT database.

17.4 Tools and documentation

In this section, we provide information about tools and documentation that may help you complete this task.

17.4.1 Tools

A number of tools are available in OPC that can assist with the migration of data:

- Batch Command Interface Tool (BCIT)
- Batch Loader
- Program Interface (PIF)

Batch Command Interface Tool (BCIT)

The BCIT can be used to unload the AD and OI databases. The nice part about using this tool to unload the databases is that the database records are unloaded into Batch Loader-format statements. These statements can then be used as input to the Batch Loader to load the entries into the target OPCplex. A sample batch job called BCIT is provided in the Additional Materials for this redbook. Each run of the BCIT job will only unload one database (either the AD or the OI, depending on the BCIT control statements used).

Before running the BCIT job, you will need to change the data set referenced on the EQQMLIB DD statement to point to your SEQQMSG0 data set. You must also change the PARM statement on the EXEC card and replace it with the name of the OPC controller subsystem that owns the database that you wish to unload.

The BCIT job unloads the AD and OI database into the data set referenced on the BATCHL DD statement. The LRECL of this data set is 80 and the RECFM is FB.

Once the data is unloaded, you can use ISPF EDIT to make any required changes to the Batch Loader statement before using the sample BATCHL job to load these entries into the target OPCplex.

The BCIT is documented in *Tivoli Workload Scheduler for z/OS Programming Interfaces,* SH19-4545.

Batch Loader

The Batch Loader uses statements to load data directly into OPC's AD and OI databases. This tool can be used as an alternative to the OPC ISPF dialogs, and is especially suited to making large numbers of changes, as in when merging databases. When used in conjunction with the output from the BCIT, this allows you to easily unload and load the AD and OI databases.

A sample job called BATCHL is provided in the Additional Materials for this redbook to run the Batch Loader. Before running the job, you will need to change the data sets on the following DD statements:

- EQQMLIB - Change to your OPC message data set SEQQMLIB.
- EQQWSDS - Change this to your target sysplex OPC WS database.
- EQQADDS - Change this to your target sysplex OPC AD database.
- EQQOIDS - Change this to your target sysplex OPC OI database.

Also included in the sample on the SYSIN DD statement are the following statements:

```
OPTIONS
  ACTION(ADD)
  SUBSYS(TWSC)
  CHECK(Y)
  ADID(EBCDIC)
  OWNER(EBCDIC)
  MSGLEVEL(2)
```

These statement are required once, at the beginning of a set of Batch Loader statements. They provide the defaults. Most important is the SUBSYS name; this must be changed to your target sysplex OPC controller subsystem name. The other statements can remain as they are.

Note: The OPTIONS statement has been included in the BATCHL job because the BCIT does not produce an OPTIONS statement in its output.

When you are ready to reload your AD or OI data into the target OPCplex, use this sample job to submit your Batch Loader statements.

The OPC subsystem must be active when the Batch Loader job runs.

The Batch Loader is documented in *Tivoli Workload Scheduler Planning and Scheduling the Workload,* SH19-4546.

Program Interface samples

There are four sample OPC PIF REXX execs provided in the Additional Materials for this redbook. One is called UNLOAD and it does an unload of a database, while another is called RELOAD and it does a reload of the database. These samples can be used on the databases that cannot be unloaded/reloaded using the standard OPC-supplied tools.

For each of these databases, the UNLOAD exec unloads the database to a sequential file. This file can be edited to make any required changes or to delete any records that do not need to be merged. This file is then used as input to the RELOAD exec which loads the data into the corresponding target database.

The calendar, period, special resource and workstation databases have variable length records. The output file used by the UNLOAD/RELOAD execs to hold the unloaded data for editing uses the maximum record length of any record in these databases, which is 32000 bytes.

The JCL Variable database also has variable length records; however, it has a maximum LRECL of 131072. Because the largest sequential data set you can edit using ISPF is 32760 bytes, an additional REXX exec called JCLCLREC is provided to identify all JCL variable tables with a LRECL exceeding 32000. These entries must be merged manually by using the OPC ISPF dialog. The UNLOAD and RELOAD execs can then be used to merge the remainder of the JCL variable tables.

The ETT database does not have variable length records. It's maximum LRECL is 200. However, to keep the allocation of output files consistent for each database, it also uses an LRECL of 32000.

The record format (RECFM) for the output file in every case is FB.

There are two sample batch jobs provided that are used to run the sample execs:

- BATCH1 is used to run the JCLCLREC exec.

- BATCH2 is used to run both the UNLOAD and RELOAD execs for each database, and the SRCPUNLD exec for Special Resources in the Current Plan.

The OPC Controller subsystem must be active when these execs are run.

You must review the names of the OPC libraries in the BATCH1 and BATCH2 jobs. Before running the jobs you need to:

1. Update the EQQMLIB DD statement to point to your OPC message library
2. Update the SYSEXEC DD statement to contain the name of the library that holds the sample execs

In addition, the BATCH2 job requires the following changes:

1. Update the CL, ETT, JCLV, PR, SR, CSR, and WS DD statements to point at a data set that will hold the offloaded definitions for associated database.
2. Update the PARM on the EXEC card to identify which exec you are running:
   - PARM='UNLOAD' for the unload exec.
   - PARM='RELOAD' for the reload exec.
3. Update the SYSTSIN DD statement to identify:
   a. The name of the target OPC subsystem. For the UNLOAD exec, this will be the name of the OPC subsystem on the incoming system. For the RELOAD exec, it will be the name of the test or the target OPC subsystem.
   b. Which database the exec will access:
      - CL for the calendar database
      - ETT for the event triggered tracking database
      - JCLV for the JCL variable database
      - PR for the period database
      - SR for the special resource database
      - WS for the workstation database
      - CSR for special resources in the CP database
   c. The name of the records that you want to unload. If you want to unload all records from the database, just specify *. If you want to unload all records whose name starts with FRED, specify FRED*. And if you just want to unload a single record, and the name of that record is HARRY, specify HARRY.

The supplied sample execs, and the functions they provide, are as follows:

- JCLCLREC is used to identify which JCL variable tables have an LRECL greater than 32000.
- UNLOAD is used to unload the definitions from an OPC database into the output sequential file identified by the DD statement with the same DDNAME as the resource of the database it is unloading.
- RELOAD is used to reload the OPC definitions from the sequential file identified by the DD statement with the same DDNAME as the resource of the database it is reloading into the target OPCplex database.
- SRCPUNLD is used to create a job that will issue SRSTAT commands to define Special Resources in the target OPCplex CP.

Record Layout of REPORT1 File

The UNLOAD exec unloads the referenced database into a sequential file, where each line represents one database record. Each database record contains two distinct parts: the first part of the record is called the *header* section, and the second part is the *data* section. The header section describes all the parts in the data section. All you really need to know when editing the REPORT1 file is where the data section starts.

The easiest way of doing this is by looking at the file. Each header is 16 bytes long. The first eight bytes contain the name of a database segment, while the second eight bytes contain the offset to where the data for this segment is in the record. There can be one or more headers.

To determine where the headers end and the data starts, the last header record has blanks in the first eight bytes. So you can simply scroll right across the file until you see the blanks.

The data part of the record starts after this header. You should be able to recognize the data, as it will have the name that you are planning to change. This can be seen in the following example:

```
000001 WSCOM           WSWD       {    WSIVL      0                    CPU1 CAN
000002 WSCOM           WSWD       {    WSIVL      0                    GZ   GSN
000003 WSCOM      Ø    WSWD            WSIVL           WSWD       &    WSWD
000004 WSCOM           WSWD       {    WSIVL      0                    JCL1 GSY
```

Records 000001, 000002, 000004 all have three headers: WSCOM, WSWD. and WSIVL. After WSIVL, you can see the blank eight bytes of the next header, signifying the end of the headers. Immediately after this, on record 000001, you can see the name CPU1. This is the name of a workstation and is therefore the data you will probably be editing.

Record 000003 has five headers that you can see in the example: WSCOM, WSWD, WSIVL, WSWD, and WSWD. You would need to scroll right until you find the blank header to find the data portion of this record.

Once you have identified the name of the record, you can edit it or delete the entire record if required. You can then save the file and use the appropriate RELOAD exec to load the data into the target OPCplex.

17.4.2 Documentation

All of the supporting documentation you should need can be found in one of the following manuals:

- *Tivoli OPC Customization and Tuning,* SH19-4380
- *Tivoli Workload Scheduler for z/OS Programming Interfaces,* SH19-4545
- *Tivoli Workload Scheduler Planning and Scheduling the Workload,* SH19-4546

18

System Automation for OS/390 considerations

In this chapter, we discuss the following aspects of moving a system that is using System Automation for OS/390 into a sysplex containing systems that also run System Automation for OS/390:

- Is it necessary to merge the System Automation for OS/390 environments for all the systems, or is it possible to have more than one System Automation for OS/390 environment in a single sysplex?
- A checklist of things to consider should you decide to merge the System Automation for OS/390s.
- A list of tools and documentation that can assist with this exercise.
- General automation considerations not specific to any particular automation software.

We also cover other, more general, automation considerations.

Note: The guidelines in this chapter are based on System Automation for OS/390 Version 2.1. As a result, some of the points we raise may not apply to earlier versions or releases of this product.

18.1 One or more SA/390 environments

In line with the merge methodologies described in 1.2, “Starting and ending points” on page 2, this chapter has been structured to provide considerations for the BronzePlex, GoldPlex, and PlatinumPlex environments. While from a sysplex point of view, these may not directly correlate to System Automation for OS/390, the scope is defined as follows:

BronzePlex merge of System Automation for OS/390
The incoming system being merged in to the target sysplex would be done so essentially to obtain the benefit of reduced software charging; therefore, from an System Automation for OS/390 perspective, there is no need to merge the automation environment of the incoming system with that of the target sysplex. In this case there are very few changes, the main consideration being the impact the introduction of a sysplex environment will have on the incoming system.

GoldPlex merge of System Automation for OS/390
The incoming system would typically be set up to share the same system software environment (sysres and program product libraries). From an System Automation for OS/390 perspective, this covers changes to the incoming system System Automation for OS/390 environment to accommodate sharing of NetView and System Automation for OS/390 install and common libraries with the target sysplex.

PlatinumPlex merge of System Automation for OS/390
In a PlatinumPlex, there would typically be sharing of all resources, and full workload sharing. Therefore, it is important that the automation should be set up to behave consistently across all systems, with a centralized point of control for all systems in the sysplex. Therefore, the incoming system’s environment would be converted to a procedural NetView, with the System Automation for OS/390 Focal Point being on another system in the target sysplex.

18.2 Checklist of considerations for merging SA/390

Table 18-1contains, in checklist format, a list of things that must be considered should you decide to merge two or more System Automation for OS/390 environments.

Table 18-1 Considerations for merging SA/390s

Consideration	Note	Type	Done
Suppress or automate new messages introduced by sysplex.	**1**	**B, G, P**	
Ensure that SA/390 and NetView are set up to only automate messages from the system on which they are running.	**2**	**B, G, P**	
On the incoming system, commands issued by System Automation for OS/390 can be directed to other system(s).	**3**	**B, G, P**	
Use EMCS consoles to avoid impacting the sysplex limit of 99 MCS consoles.	**4**	**B, G, P**	
No changes to System Automation for OS/390 automation during system IPL or system shutdown are required to accommodate the move to sysplex.	**5**	**B, G, P**	
It is possible to have different System Automation for OS/390 subplexes within the same sysplex.	**6**	**B, G, P**	
Migrate the incoming system’s ACF into an enterprise PDB™.	**7**	**B, G, P**	

You will need to work closely with your systems programmers and the other support teams involved in the merge in order to ensure that any environmental changes have corresponding changes to automation policy as identified.	**8**	**B, G, P**	
Review message suppression, either MPFLST and/or Automation-controlled.	**9**	**G, P**	
Consider the usage of system symbolics to help reduce the installation and maintenance effort.	**10**	**G, P**	
Consider changing your NetView domain name(s) to follow the standard implemented on the target sysplex.	**11**	**G, P**	
Consider implementing a NetView (including System Automation for OS/390) data set naming standard in line with that used on the target sysplex.	**12**	**G, P**	
Consider implementing NetView install, common, and domain specific library structures.	**13**	**G, P**	
Consider using a single NetView System Automation for OS/390 procedure, set up to use symbolics to differentiate systems, data set names etc.		**G, P**	
Sharing a sysres may mean changes to the way SMF post-processing is managed and automated.	**14**	**G, P**	
Determine if you are to use ARM or System Automation MOVE groups to restart applications when moving into the target sysplex automation environment, as this is likely to mean changes to the incoming system's System Automation for OS/390 and possibly ARM definitions to accommodate.	**15**	**G, P**	
If you're going to be sharing a common system software environment, then you will need to consider any additional System Automation for OS/390 features such as CICS, IMS and OPC Automation.		**G, P**	
We recommend that you review the incoming system's subsystem extensions so that they follow the standard implemented on the target sysplex.	**16**	**P**	
Review and change the incoming system's NetView security settings in preparation for merging with the target sysplex automation environment.	**17**	**P**	
Automation Manager Considerations in SA/390 V2R1.	**18**	**P**	
From SA/390 V2R1, the Automation Manager (AM) uses the system logger to record and retrieve its actions and messages. This will require CF and Logger definitions.			
Define the incoming system's automation objects to establish it as a new target to an existing System Automation for OS/390 focal point.	**19**	**P**	
If you are looking at redistributing the incoming system's application load across other members of the target sysplex (either for load balancing or in case of outage), then you will need to consider what the automation implications will be.		**P**	

The "Type" specified in Table 18-1 relates to the sysplex target environment—**B** represents a BronzePlex, **G** represents a GoldPlex, and **P** represents a PlatinumPlex.

Notes indicated in Table 18-1 on page 292 refer to the following points:

1. Messages generated in a Parallel Sysplex environment need to be considered before the incoming system is merged into the target sysplex. These additional messages will need to be suppressed, displayed, automated, or given special consideration in line with your existing automation standards.

   Assuming that this issue has already been addressed within your existing target sysplex environment, then it is reasonable to assume that you would want to include the same levels of automation on the incoming system.

   Another approach would be to collect a number of days of SYSLOG data and analyze it using a program such as the MVS SYSLOG Message Analysis Program, which can be downloaded from:

   http://www.ibm.com/servers/eserver/zseries/software/sa/sadownload.html

   An excellent source of information on automation in a Parallel Sysplex environment, including details on which messages should be suppressed, automated, and so on, is the IBM Redbook *Parallel Sysplex Automation Guidelines,* SG24-5441.

   In addition, System Automation for OS/390 provides sample MPF members and MATs within the SINGSAMP library.

2. In a Parallel Sysplex environment, messages are routed to all active consoles on all systems that are eligible to receive a particular message.

   Note: MPF processing has system scope; thus, you must plan MPF processing on each system in the sysplex. The system default makes all messages eligible for automation. For details of the MPFLSTxx options and parameters, refer to *z/OS MVS Initialization and Tunning Reference,* SA22-7592.

   NetView receives its messages from the Subsystem Interface (SSI) or EMCS console(s), depending on your NetView configuration. The %AOFDOM% synonym declared in AOFMSGSY is used to protect NetView from actioning messages that did not originate on the system on which it is running.

3. In a sysplex environment, commands can be routed to other systems within the sysplex.

   NetView will only issue commands to the system on which it is running. If you need to issue commands to other systems, this can be accomplished through the use of the system (MVS) ROUTE command or the NetView RMTCMD command. The RMTCMD command sends system, subsystem, and network commands to another NetView address space within your network.

4. You should use EMCS consoles in NetView, which can be set up as follows:
   - Assign each CSSIR task a unique TSKID name within the sysplex. To do this, CNMCSSIR needs to be replaced with a unique name in DSIDMN, CNME1035, DSIMSG00, and DSIMSG01.
   - Each unique SSIR task name should be coordinated between DSIDMNK, AOFMSGSY, and CNME1035 (NetView 1.3 only).
   - Specify MSGIFAC=SYSTEM on both the NetView task (in DSIDMNK) and the SSI task (in the procedure itself).
   - Add the AOCGETCN command to the initial CLIST of your operator profiles. The AOCGETCN command provides a common global variable named AOFCNMASK to obtain unique console IDs. AOFCNMASK is described in Appendix B of *System Automation for OS/390 Customizing and Programming,* SC33-7035.

5. In a sysplex environment, additional messages (XCF WTORs) are displayed at IPL time when joining the sysplex, and also during shutdown when a system is leaving a sysplex.

These WTOR messages cannot be automated because System Automation for OS/390 is not yet active.

If you are using Processor Operations (or ProcOps, as it is commonly referred to), it is recommended that you add the incoming system target processor hardware to your ProcOps definitions on your automation focal point. ProcOps provides a single point of control to power off/on, reset, perform IPLs, set the time of day clocks, respond to messages, monitor status, and detect and resolve wait states.

If you are not using ProcOps, we recommend that Sysplex Failure Management (SFM) be used to avoid XCF WTOR messages.

For more information on ProcOps customization, refer to:

- Chapter 4 of the IBM Redbook *Parallel Sysplex Automation: Using System Automation for OS/390,* SG24-5442
- *System Automation for OS/390 Planning and Installation,* SC33-7038

6. Within System Automation for OS/390, you can specify a group id to define a subset of systems within the sysplex environment. The logical sysplex group ID may be 1 or 2 alphanumeric or national characters, and is specified on the GRPID parameter in HSAPRM00.

   All of the systems that you want to place in the same subplex should specify the same GRPID value. This value will be prefixed with the string INGXSG to construct the XCF group name. To determine existing XCF group names, you can use the **`D XCF,GROUP`** system command.

7. We recommend that you use a single System Automation for OS/390 PDB within your enterprise automation environment. ACFs are sharable within a sysplex, therefore, a build of the PDB will create an ACF containing all relevant system definitions.

   For System Automation for OS/390 V2R1, a PDB build will create an ACF and an Automation Manager Configuration (AMC) file. Both files must be shared between all System Automation for OS/390 and Automation Managers within your automation sysplex.

   There is no utility to merge ACFs from different policy data bases. You must manually compare the resources defined in each PDB, and make appropriate changes to your target sysplex “enterprise” PDB to add any resources that are only defined in the incoming system.

   An overview of the steps to accomplish the merge to a single enterprise PDB could be as follows:

   - Before you start adding definitions for the incoming system, we recommend that you take a backup of your target sysplex PDB.
   - Define the incoming system to the target PDB, using the definitions from the incoming system’s PDB as a model.
   - Connect the incoming system (defined above in the target PDB) to the sysplex group.
   - Define a new group (for example, “incoming system”) to contain all applications to be defined for the incoming system:
     - Applications that are the *same* or *common* between the incoming system and target sysplex need to be connected to the application group associated to the incoming system.
     - Applications with *different* requirements between the incoming system and target sysplex environments must first be defined within the target PDB, and then connected to the application group associated to the incoming system.

These steps are in line with the normal automation policy definition process. For more information, refer to *System Automation for OS/390 Defining Automation Policy,* SC33-7039.

- Any MVS messages that are unique to the incoming system must be added to the target sysplex PDB.
- If you are not sharing the JES2 spool, then the JES2-defined resources, including spool recovery, should be duplicated on the target PDB from existing definitions already within the incoming system's PDB.
- Add any unique user-entry definitions for the incoming system.
- Perform an ACF Build and load the Automation Manager into the target sysplex environment. Once loaded, INGLIST can be used to verify the correct applications and their relationships to the incoming system. This is possible without having to first bring the incoming system into the target sysplex, as the information is loaded into the Automation Manager.

8. Depending on your level of automation, and how much change is being made to the incoming system based upon what resources you will be sharing with the target sysplex, you will need to carefully consider any accompanying automation policy changes required in order to accommodate these changes.

   For example, if you currently have some JES2 automation in place, this may need to be reviewed if JES2 on the incoming system was to become part of a MAS (shared spool environment with the other target sysplex systems). Other environmental changes that may affect your automation policies could be LPAR, SMF id, NJE, Initiators, Printers, VTAM Domain, HSM, Subsystem names, Job names, DASD and other device definitions, and so on. It is important that you work closely with your systems programmers and the other support teams involved to ensure that issues are identified and addressed.

9. In preparation for sharing a common sysplex-wide Parmlib, you should review your MPFLSTxx definitions and automation message management to conform to a single standard. Another option would be to set up separate MPFLSTxx members in a common Parmlib for use by individual systems within the sysplex.

   Note: Given a choice, we recommend the use of a common MPFLSTxx member on all systems in the target sysplex.

10. NetView has supported the use of MVS system- and user-defined symbols since NetView V1R1. For more information on how to use symbols in a NetView environment, see the IBM Redbook *Automation Using Tivoli Netview OS/390 V1R3 and System Automation OS/390 V1R3,* SG24-5515.

11. Although it is not a requirement, unless you have a domain name conflict within the target sysplex, we recommend that you implement a standard NetView domain name standard and convert the incoming system to conform to it.

12. In a GoldPlex or PlatinumPlex, where the sysres and other systems software libraries will be shared sysplex-wide, we recommend that you implement a NetView automation data set naming standard in line with that used on the target sysplex.

    Note: Consider the impact of changing data set names on facilities such as the System Automation Policy Database dialog, in that you will need to review and modify any TSO logon procedures or allocation methods to reflect data set name changes.

13. In order to reduce the effort and time required to implement and maintain your NetView and System Automation for OS/390 environment sysplex-wide, we recommend that you implement a standard library structure of Install, Common and Domain-specific libraries across all participating members of the sysplex. For more information, see Chapter 4 of the IBM Redbook *Automation Using Tivoli Netview OS/390 V1R3 and System Automation OS/390 V1R3,* SG24-5515.

14. In a GoldPlex or PlatinumPlex, where the target sysplex Parmlib and systems software environment is to be shared with the incoming system, you will need to review and possibly modify your SMF post-processing automation (via in-house written automation routines, or via SA's own SMF dump automation policy, or by using the SMF post-processing exit (IEFU29)).

15. Automatic Restart Manager (ARM) is an MVS function designed to provide efficient restarts of critical applications when they fail. System Automation for OS/390 is able to track functions performed by ARM. For more information, see section 3.5 of the IBM Redbook *Parallel Sysplex Automation: Using System Automation for OS/390,* SG24-5442.

16. If you are going to be running either a network NetView, or an automation NetView, or both, you should implement a standard for system subsystem extensions (defined in the IEFSSNxx member in parmlib) in order to ensure unique names in a shared sysplex or shared automation environment.

    You should also consider, and implement, a standard to ensure unique SSIR tasknames, which is best accomplished through the use of system symbols; for example, change the following statement in member AOFMSGSY of your common NETV.DSIPARM library: **`SYN %AOFSIRTASK% = ‘’’&DOMAIN.SIR’’’`**

17. NetView security can be set up to use SAF for your external security product (CMDAUTH=SAF), or to use NetView's own security, using the NetView command authorization table (CAT).

    If the incoming system and target sysplex environment both use CAT, you will need to review both environments in order to ensure the required level of access is maintained across your automation sysplex.

    However, if the incoming system and target sysplex automation environments use a combination of CAT and SAF, then you will presumably want to convert the incoming system to the standard already in place on your target sysplex. You will, of course, also need to consider your incoming system and target sysplex's security definitions, and make changes as necessary in order to ensure the required level of access is maintained across your automation sysplex.

    For more information on converting between the different System Automation for OS/390 security methods, see "Scenarios for Converting Types of Security" in the publication *Tivoli Netview for OS/390 Security Reference,* SC31-8606.

18. For System Automation for OS/390 V2R1, there is now the concept of the Automation Manager (AM) and the Automation Agent (AA). The AM runs as a separate address space, where you will need at least one primary AM with possibly one or more backups. The AM effectively acts as the automation decision server to the AAs within the automated environment.

    Moving from a single system to a sysplex-wide automation environment increases the need for continuous availability. You may want to consider using the incoming system as a secondary AM environment that would be used to take over the primary AM function in the event of a failure or scheduled outage. In such a scenario, both the primary and any secondary automation managers must have access to the following:

    – Shared DASD, containing:
      - The configuration data (result of the ACF and AMC build process)
      - The configuration data (schedule overrides VSAM file)
      - The configuration information data set, which is a mini file in which the Automation Manager stores the parameters with which to initialize the next time it is HOT or WARM started.

– All automation agents must have access to the CF. If you're using MQSeries for communication, then the following queues are used by SA/390:
  - WORKITEM.QUEUE for the Workitem Queue
  - STATE.QUEUE for the Automation State Queue
  - AGENT.QUEUE for the Automation Agent Queue

It is expected that the incoming system's System Automation for OS/390 environment would be set up as an AA, communicating to your AM. For communication between AAs and the AM, you need to coordinate the GRPID statement in the AMs Parmlib member (HSAPRMxx), with the INGXINIT member defined in the DSIPARM of each associated AA. Assuming you are already running a similar setup within your target sysplex environment anyway, then none of this should be too unfamiliar.

For more information, see *System Automation for OS/390 Planning and Installation,* SC33-7038.

19. Although it would be more normal to have a single automation focal point when running in a PlatinumPlex, it is also possible that you would want to avail of this capability in a GoldPlex or even (although less likely) in a BronzePlex. To achieve a common focal point, remember that you will need to take the following actions:
   – Define gateway connections and forwarding definitions in the target sysplex PDB to connect the new system to the focal point.
   – If you are using RACF or another SAF product for userid validation, the gateway operators must be defined to the appropriate security subsystem.

18.3 Tools and documentation

In this section we provide information about tools and documentation that may help you complete this task.

18.3.1 Tools

- SYSLOG Message Analysis Program, which can be downloaded from:

  http://www.ibm.com/servers/eserver/zseries/software/sa/sadownload.html

18.3.2 Documents and References

- *z/OS MVS Initialization and Tunning Reference,* SA22-7592
- *System Automation for OS/390 Defining Automation Policy,* SC33-7039
- *System Automation for OS/390 Planning and Installation,* SC33-7038
- *System Automation for OS/390 Customizing and Programming,* SC33-7035
- *Automation Using Tivoli Netview OS/390 V1R3 and System Automation OS/390 V1R3,* SG24-5515
- *Parallel Sysplex Automation: Using System Automation for OS/390,* SG24-5442
- *Tivoli Netview for OS/390 Security Reference,* SC31-8606
- http://www.ibm.com/servers/eserver/zseries/software/sa

18.3.3 Group forums

The following Yahoo Groups (forums) are also available should you wish to exchange information and ask questions of other experts in the field:

- http://groups.yahoo.com/group/SAUsers/
- http://groups.yahoo.com/group/Netviews/

19

Operations considerations

There are many operational considerations when moving a system into a sysplex. This chapter discusses the considerations for Hardware Management Console (HMC) setup, console setup and management, and other general operations procedures.

In this chapter, we will discuss the following:

- Should you merge your HMCs?
- What do you need to consider when planning a merge of HMCs?
- Should you merge consoles?
- What do you need to consider when planning a merge of consoles?
- General operational considerations when merging a system(s) into a sysplex environment.

We will also provide:

- A checklist of items to consider, should you decide to merge your HMCs and/or consoles.
- A methodology for carrying out the merge of the HMCs and/or consoles.
- A list of tools and documentation that can assist with these tasks.

19.1 One or more HMCplexes

It is possible to have more than one HMCplex in a single sysplex. Similarly, you can have more than one sysplex in an HMCplex. The definition of an HMCplex is “the set of HMCs that all manage the same set of processors”.

An HMC does not have to map to a sysplex—in fact, it would be very unusual if all the LPARs in all the processors managed by a HMC were all in the same sysplex. An HMC can also manage systems that are stand-alone or spread across multiple sysplexes, provided that they are all in processors that are in the same HMC Management Domain as the HMC. The HMC Management Domain provides a method for you to maintain the security of a processor by controlling the access of the HMCs to the CPC[1] Support Element (SE).

HMCs can only communicate with CPC SEs that have the same domain name and domain password as the HMC. Assigning a unique domain name and password to a HMC and the CPCs that are defined to it will isolate those CPCs from any other HMC connected to the same Local Area Network (LAN). It is recommended that you assign a unique domain name and password if the SEs will be attached to a non-private (for example, company-wide) network.

For more information on HMC Management Domains, see *Hardware Management Console Operations Guide,* SC28-6809.

To assist in the decision about whether to merge all the systems into a single HMCplex, we have defined three sysplex configuration environments, BronzePlex, GoldPlex and PlatinumPlex, in 1.2, “Starting and ending points” on page 2.

Assumption: In this section we assume that the CPC that the incoming system is running on is *not* managed by the same HMC(s) as the CPCs that the target sysplex systems are running on.

If this is *not* the case, then you do not have to do any work in regard to moving to a single HMCplex. However, you may still find it worthwhile to read the remainder of this section for general hints and tips about how to set up and maintain your HMCplex.

If your target is a BronzePlex environment, you could maintain a separated HMCplex environment without merging any of the HMCs (this assumes that the incoming system is running on a CPC that is not currently in the same HMCplex as the systems in the target sysplex). The incoming system would continue to have its own HMC, but this configuration would not offer any benefits in terms of ease of operational management. Operations personnel would be required to operate the CPCs from separate physical HMCs even though the HMCs may be co-located in the same physical desk area.

A more manageable approach in a BronzePlex, compared to multiple HMCplexes, might be to set up separate HMC userids—one set that can manage the incoming system’s LPAR(s), and another set that can manage the target sysplex’s LPARs. This way you can protect who can manage each system, while avoiding the overhead of maintaining more HMCs than necessary.

[1] In this IBM Redbook, you may see the term Central Processor Complex (CPC) or Central Electronic Complex (CEC) being referenced. These terms are inter-changeable and can be used to describe a hardware environment which consists of central storage, one or more central processing units, time-of-day clocks, and channels which are, or can be, placed in a single configuration. A CPC or CEC also includes channel subsystems, service processors and expanded storage, where installed.

For the GoldPlex and PlatinumPlex environments, merging the HMCs would provide the benefits of a consolidated operational environment. The daily operation and management of multiple CPCs would be simplified, and a single operational focal point would be established.

19.1.1 Considerations for merging HMCplexes

Table 19-1 contains, in checklist format, a list of items that must be considered should you decide to merge two or more HMCplexes.

Table 19-1 Considerations for merging HMCplexes

Consideration	Note	Type	Done
If you merge the HMCplexes, will you have more HMCs in the HMCplex than you require?	1	B, G, P	
If you are going to eliminate some HMCs, which ones should you eliminate?	2	B, G, P	
What is involved in “merging” HMCs?	3	B, G, P	
Review physical and network security considerations for the HMCs - does the HMC have a requirement to be connected to a dedicated private LAN?	4	G, P	
Ensure HMCs are at the required LIC level - a down-level HMC cannot support an up-level processor.	5	G, P	
Ensure all required CPCs are accessible from all HMCs in the HMCplex.	6	G, P	
Review HMC backup and recovery procedures.	7	B, G, P	
Check the HMC microcode delivery and load procedures used by the hardware representative.	8	B, G, P	
Review Change Management procedures.	9	B, G, P	
Is the HMC to support additional applications, such as controlling an ESCON Director or Sysplex Timer Model 2?	10	B, G, P	
Review HMC remote operations options.	11	B, G, P	
Enable the HMC Web interface and configure relevant networking definitions.	12	B, G, P	
For remote operations, is a HMC required or will the HMC Web interface be sufficient?	13	B, G, P	
Group the incoming system(s) into the sysplex group with the other existing systems in the sysplex.	14	G, P	
Update Image profile for the Initial and Reserved values for processors.	15	G, P	
Update Load profile for incoming system to ensure the correct sysres and load parameters are specified.		G, P	
Specify DEVNUM(SYSCONS) in the CONSOLxx member of parmlib to allow the HMC to be given a NAME and AUTH value.	16	B, G, P	
Review Initialization Message Suppression Indicator (IMSI) value of the Load Parm field in the LOAD Profile.	17	B, G, P	

Consideration	Note	Type	Done
Use of V CN(*),ACTIVATE command to utilize the HMC console when needed for recovery.	**18**	**B, G, P**	

The “Type” specified in Table 19-1 on page 303 relates to the target sysplex environment—**B** represents a BronzePlex, **G** represents a GoldPlex, and **P** represents a PlatinumPlex. In the table, if the type column contains a “B”, this represents an environment where you have decided *not* to merge the HMC for the incoming system with the HMCplex controlling the target sysplex. For example, you only need to review the physical and network security considerations for the HMCs (fourth row) if your target environment is a GoldPlex or a PlatinumPlex (because you will not be making any changes in this area). However, we *do* recommend reviewing your backup and recovery procedures regardless of what your target environment is (Type column contains a B, a G, and a P).

Notes indicated in Table 19-1 on page 303 are described below:

1. Perhaps the first question we should look at is what is the “ideal” number of HMCs in a HMCplex? The answer, of course, is that “it depends” on your operational setup. First, what are the limitations:
   - Any given HMC can manage no more than 100 CPCs.
   - Any given CPC can simultaneously be managed by no more than 32 HMCs.
   - Any given CPC must be managed by at least one “local” HMC (“local” means the HMC and SE are on the same physical LAN). The LAN can be either Token Ring or Ethernet (assuming the CPCs and HMCs are at a level that support Ethernet (HMC and SE 1.4.0 and later)).

   Now, the “ideal” number of HMCs really depends on how many locations you want to directly manage your CPCs from. As a rule of thumb, we suggest two (for redundancy) “local” HMCs (that is, on the raised floor in reasonable proximity to the CPCs), and two more (for redundancy) HMCs at each operations center (that is, where the operators sit).

   The HMC Web interface can also be used from other locations, but we don't normally recommend this be used as the full time method of management. The Web interface is intended for occasional use by users such as system programmers, or as a backup method of management.

   So, if you have one computer room, and all your operators are in one location outside the computer room, then the ideal number of HMCs would be four. If merging the incoming system HMC(s) into the target sysplex HMCplex would result in more than four HMCs in this environment, then it may be better to just add the security definitions from the incoming system HMC(s) to the target sysplex HMCs, and eliminate some of the HMCs as part of the merge process.
2. If you are going to eliminate some HMCs as part of the merge, you should eliminate the oldest or the most poorly configured ones. As new levels of support are announced, they typically have more requirements in terms of MHz, disk space, memory, and so on, so keep the newest HMCs that have the largest configurations.
3. If you are going to merge HMCplexes, you will need to change each HMC so that they all have the same definitions. This means probably changing the Management Domain of some of the HMCs and also that of the CPC that the incoming system runs on (so that it can be in the same Management Domain as all the target sysplex CPCs).

You should also have the same security definitions (userids and the groups and actions they have access to) on all HMCs in the HMCplex. At the time of writing, there is no automatic way to do this. If the incoming system HMCs have userids that are not defined on the target sysplex HMCs, you will need to add those definitions manually. Similarly, if you are going to move the incoming system HMC(s) into the HMCplex, you will need to add the security definitions from the target sysplex HMCs to the incoming system HMC(s)

4. If security is a major concern for your environment, connect all HMCs and CPCs onto a private LAN. The use of a private LAN offers security as follows:
   - Direct access to the LAN is limited to the HMCs and CPCs connected to it, so there is no possibility of anyone trying to "hack" the HMCs or SEs.
   - There is no possibility that a faulty user PC or faulty card in a user PC could disrupt the HMC LAN.

   To get the required security from such a configuration, do the following:
   - Assign a unique domain name that includes all the CPCs controlled from the HMCplex.
   - Create userids and passwords for the HMCs and change the passwords for the default userids.
   - If using the HMC Web interface, ensure that only the appropriate people are given access.
5. All HMCs in the HMCplex must support the highest level CPC that is accessible from that HMCplex. If any of the HMCs do not support that level, those HMCs will only be able to manage a subset of the CPCs, and therefore not provide the level of redundancy that you need. The driver level on the HMC will need to be verified, as a down-level HMC cannot support an up-level processor. Your hardware CE can help with this task.
6. Ensure that the HMCs that are going to make up the HMCplex are able to access all of the required CPC SEs. This is because the CPC SEs can be on completely different networks and may even be in different locations. However, even if the CPCs are remote, they can still be managed from the same HMC in a centralized location.

   You will also need to be aware of the HMC and CPC SE network addresses (TCP/IP), Network names (SNA) and CPC object names.
7. If your existing backup and recovery procedures have been tested and are satisfactory, there should be no need to change the procedures even if you decide to merge the HMCs into a single HMCplex. All the HMCs are peers of each other, so all should be backed up in a similar manner.
8. The HMC microcode delivery and load procedures used by the hardware representative should not change if you are merging HMCs into the HMCplex.
9. If all HMCs in the HMCplex have access to all the SEs (which they should have), and the SEs are running Version 1.6.2 or higher, then all the HMCs in the HMCplex should have "LIC change" enabled. This is a change in 1.6.2 that is intended to provide better availability in case of a failure in any of the HMCs.

Tip: At least one HMC in the domain must have "LIC change" enabled.

10. It is possible to use an HMC PC to manage your ESCON Directors and Sysplex Timers, in addition to its role as an HMC. Note, however that these programs run in a loop, polling the devices they manage, and as a result can use up to half the processing power of the HMC. If the HMC will be managing a large number of CPCs, we do not recommend running the ESCON Director or Sysplex Timer code on that HMC. Refer to the relevant ESCON Director and Sysplex Timer manuals for more information on the required configuration and setup tasks.

11. If you want to operate the HMC from a remote location, the recommendation is to have a fully configured HMC in the remote site. The remote HMC must be kept at the same level of LIC, or higher, than the SEs it controls. An alternative to this is to access the HMC through a Web browser.

    You can use the Web browser as a remote operation option to monitor and control a local HMC and its configured SEs. HMC 1.4.2 (originally shipped with 9672 G4 CPCs) and latest level HMCs have a Web Server built in. When enabled and properly configured, the HMC can provide a representation of the HMC user interface to any PC with a supported Web browser connected through your LAN using the TCP/IP protocol.

    Important: Security for a browser connection is provided by the HMC enablement functions, HMC user logon controls, and network access controls. Encryption of the browser data should be added by selecting the "`enable secure only`**"** selection (available with HMC version 1.6.2 and levels). This requires that the user's Web browser supports the HTTPS protocol. If this option is not selected, the userids and passwords use to logon to the HMC are sent in clear text across the network.

12. If you decide to enable the HMC Web interface, we strongly recommend that you review your security requirements prior to enabling this feature. Additional information can be found in the *Hardware Management Console Operations Guide,* SC28-6809.

13. Starting with HMC version 1.6.2, the HMC provides two types of Web browser access: "Remote entire Hardware Management Console desktop" and "Perform Hardware Management Console Application tasks". The "Remote entire Hardware Management Console desktop" type remotes the keyboard, mouse, and display of the HMC to the Web browser. Any updates at the HMC are immediately updated at the Web browser. Only one user at a time can use this type of access.

    The "Perform Hardware Management Console Application tasks" type of access sends simple HTML and Java™ scripts to the Web browser, and the HMC instructs the Web browser to automatically refresh the page contents on a fixed interval. By default this is every 2 minutes, but this can be changed from the Web browser by selecting **Console Actions -> personalize**.

    Despite this capability, we do not recommend using this interface for normal operations for the following reasons:

    – Some customers have experienced problems with Web browsers leaking memory when accessing Web pages. Since the "Perform Hardware Management Console Application tasks" relies on the Web browser refreshing the page for updates, these memory leaks add up. We found that the Web browsers would run out of memory after about 24 hours of continuous use in this mode. (However, the memory leaks are not a problem for occasional, limited time use.)

    – Only one person at a time can use the "Remote entire Hardware Management Console desktop" Web interface. This interface also uses more network bandwidth since it is remoting the contents of the display pixels. If this is not a problem for you, then continuous operation should be fine. (Note that this mode does not have the memory leak problem because the Web browser ends up running a Java applet, and thus the Web browser is not refreshing the page nor leaking memory.)

    – A real HMC will automatically re-establish communication with the SEs following a temporary network outage. A real HMC will also automatically try to use a second LAN path (if present) should the first become unavailable. In the Web browser case, someone would have to manually restart the session following a temporary network outage. This may or may not be significant depending on your requirements and expectations.

14. HMC groups contain logical collections of objects. They provide a quick and easy means of performing tasks against the same set of objects more than once. The Groups Work Area on the HMC contains the groups that a user has been authorized to. There are both system-defined groups, which are provided with the system and consist of all CPCs or all images that are in the HMC Management Domain, and user-defined groups which can be created from these groups to represent views of CPCs or images that your installation wishes to work with. Figure 19-1 shows the screen in the Customize User Controls dialog where you can specify which groups, and which object within those groups, a given user can manage. *Hardware Management Console Operations Guide,* SC28-6809 contains further information.

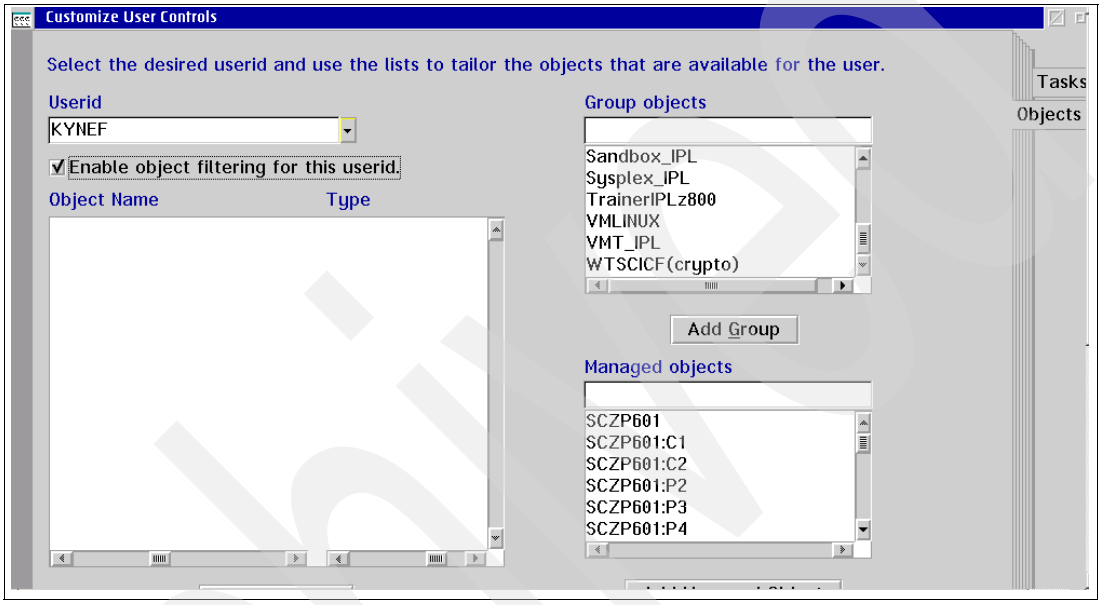


Figure 19-1 Customizing HMC user access

15. In the Image Profile, you can assign the number of logical processors to the LPAR. The LPAR can be defined with the number of logical processors (Initial + Reserved) greater than the number of processing units (PUs) currently installed. This is to cater for Capacity Upgrade on Demand (CUoD) allowing for a non-disruptive upgrade to install additional PUs. You will not need to deactivate, re-define and activate the LPAR to make use of the newly-installed PUs.

16. SYSCONS cannot be specified as a console to NIP. However, if none of the consoles specified in NIPCONS are available, then NIP automatically uses this interface. By specifying DEVUM(SYSCONS) in your CONSOLxx member, this definition can be used by every system in the sysplex because the console is automatically assigned a name equal to the z/OS system name. This console can be assigned routing codes, but in normal operation no messages are sent to it. As this interface can be rather slow, we recommend you do not use this console unless there is very low message traffic or an emergency situation.

17. There are three type of profiles (Reset, Image and Load) and they are stored on the SE. A Load Profile is needed to define the device address that the operating system is to be loaded from. Assuming the incoming system will continue to run in the same LPAR, you should not need to set up any new profiles for that system. However, you will need to review the load parameter value of the Load Profile that is assigned to the incoming system, as there may be a different Initialization Message Suppression Indicator (IMSI) character used or required.

z/OS MVS System Commands, SA22-7627 contains detailed information on the values available for the Initialization Message Suppression Indicator (IMSI).

18. You should not have the console integration support of the HMC (SYSCONS) as the only console in a sysplex. The console integration facility allows the HMC to be a NIP console (when there are no consoles available to be used during NIP as specified in your NIPCON definitions in the IODF), or it can be used as an MVS console, but only when absolutely necessary (for example, when there is a “no-consoles” condition).

    The sysplex master console is always channel-attached to a system in the sysplex. If your target sysplex configuration allows, there should be at least two channel-attached consoles to two systems in the sysplex, preferably on two different CPCs. For better operational control, it is recommended that the first system to IPL and the last system leaving the sysplex have a physically attached console.

19.1.2 Security

Since multiple HMCs and internal SEs require connection via a LAN, and remote connections are possible, it is important to understand the use and capabilities enabled for each HMC in your configuration.

HMCs operate as peers with equal access to the CPCs configured to them. The SE for each CPC serializes command requests from HMC applications on a first-come, first-served basis. There is no guarantee of exclusive control across a sequence of commands sent from a single HMC.

Security recommendations that you should consider are:

- Following the merge of the CPC, HMC and SE into your target sysplex configuration, the access administrator should verify and if necessary, change the default logon passwords at the HMC and Support Element.
- Define the HMC, Support Element and all CPCs onto a private LAN. Using a private LAN offers several security, availability and performance advantages:
  - Direct access to the LAN is limited to the HMC, Support Element and CPC. Outsiders cannot connect to it.
  - Traffic disruption due to temporary outages on the LAN is reduced - including disruptions caused by plugging in and powering on new devices, or the LAN-to-LAN adapters being run at the wrong speed.
  - LAN traffic is minimized, reducing the possibility of delays at the HMC/SE user interface.
- If a private LAN is not practical, isolate the LAN segment containing the HMC, Support Element, and CPC by providing a TCP/IP router between the isolated LAN and the backbone LAN to provide an intermediate level of isolation. Some customers have experienced problems using LAN bridges, so we recommend the use of a router instead.
- Assign a unique domain name that includes all the CPCs controlled from the HMCplex.
- Locate at least one of the HMCs in the machine room near the CPCs that form its domain.
- Establish a focal point and control access using the “enable/disable” controls provided for:
  - Licensed Internal Code (LIC) update
  - Remote service support
  - Remote customer access
  - Remote service access
  - Auto-answer of the modem

19.1.3 Optical and I/O error analysis

You should consider limiting the number of HMCs that are enabled for “Optical and I/O Error Analysis”. This will reduce duplicate error reports for errors such as the loss of an ESCON fiber link.

19.1.4 Remote access

The following items should be taken into consideration when configuring your environment for remote access to your target sysplex environment:

- Only enable modem auto-answer if there is a requirement to dial into an HMC as a means of remote operation.
- Physically locate the HMC in a secure area (for example, a locked room).
- If a remote console is used for remote operations access, assign a secure logon password.
- Logoff each HMC when not in use.
- Use the OS/2® lockup function to lock the HMC keyboard after a predetermined period of keyboard inactivity has expired.
- Customize the HMC secure desktop setting to control whether console users have access to other applications and objects on the console’s OS/2 desktop.
- Establish a limited list of objects and actions available to the operator.

19.1.5 Methodology for merging HMCplexes

The decision to merge HMCplexes could depend on several items that need to be considered:

- Physical security
  - Depending on the business workload requirements, there may need to be a high level of physical security for where you want to locate the HMCs. This could then impact on your use of certain functions within your HMCplex configuration.
- Network security
  - There may be a requirement for stringent network security requirements which may impact on the usability of certain functions in your HMCplex. For example, your business may not allow intranet access to the Web server on the HMC, and this could possibly prevent HMC Remote access via the intranet.
- Operational
  - A major benefit of having a HMCplex is that there will be a HMC focal point. This focal point will then provide operational staff a single point of control of the CPC environment.

If the decision is to merge, we recommend that the merging of the HMCplex is carried out in advance of the merge of the incoming system in order to minimize the amount of disruption. All the tasks from the checklist can be completed prior to the merge of the incoming system.

Figure 19-2 on page 310 and Figure 19-3 on page 311 show an example configuration before and after the HMCs have been merged.

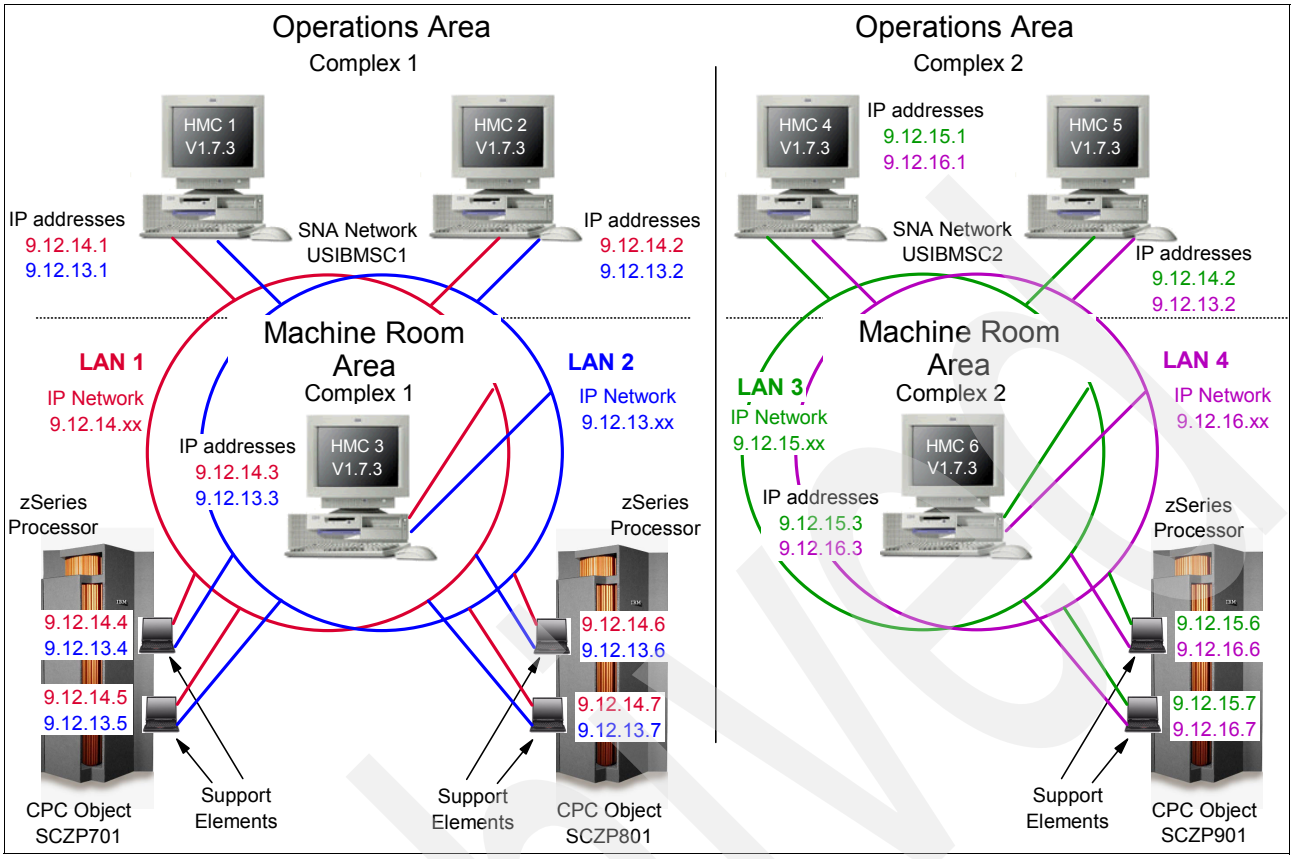


Figure 19-2 HMC configurations for target sysplex and incoming systems

Figure 19-2 illustrates the possible HMC configuration that you may have for your existing target sysplex and existing incoming system prior to the merge process. This follows our assumption that the CPC containing the incoming system is currently in a different HMCplex to the CPCs containing the target sysplex systems.

In this case, prior to the merge, the incoming system CPC is in a different machine room than the target sysplex CPCs, and the operators of that CPC have two HMCs for that CPC in the operations area. There is also one HMC in the machine room. The CPCs containing the target sysplex systems also have one HMC in that machine room, and two HMCs in the operations area. This results in a total of six HMCs.

In this example, the decision was made to place all the CPCs in the same Management Domain, and to just have a total of four HMCs managing all three CPCs. The post-merge configuration is shown in Figure 19-3 on page 311.

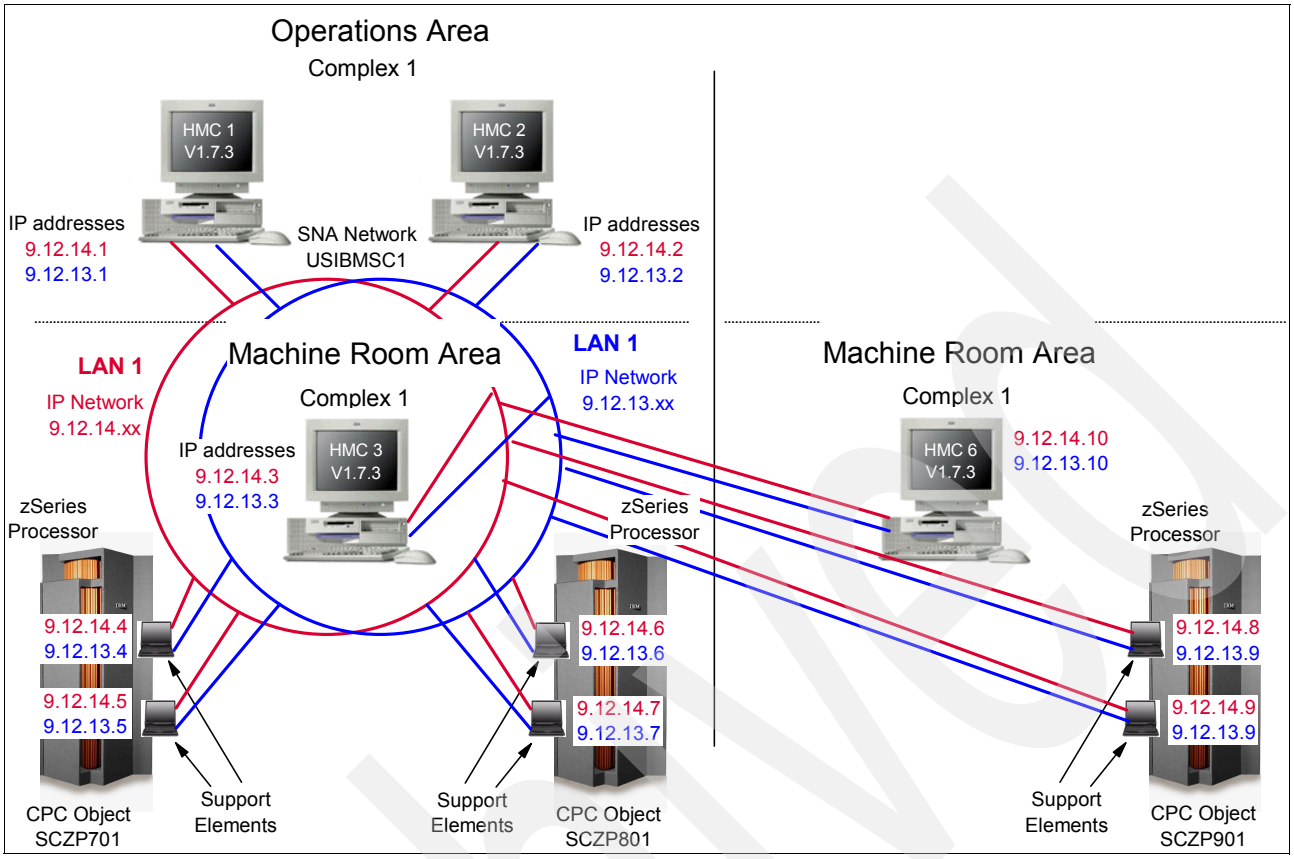


Figure 19-3 HMCplex configuration after the merge

Figure 19-3 illustrates the final HMCplex configuration after the merge has taken place.

Attention: In Figure 19-2 on page 310 and Figure 19-3, the level shown on the HMCs is significant. The version level of the HMCs must be equal to or greater than that of any CPC (SE) that they will be managing.

19.2 Consoles

Consoles are a critical resource in a sysplex. A complete loss of consoles limits an operator's ability to effectively manage the systems in a sysplex environment, and could potentially result in a system or even a sysplex outage if not addressed promptly. A correct console configuration is particularly important in a sysplex, because the way consoles are managed is fundamentally different than in a multisystem environment that is not a sysplex.

In a sysplex, there is a single pool of 99 consoles for the whole sysplex, whereas in a non-sysplex environment, *each system* can have up to 99 consoles. In addition, in a sysplex, every console can not only potentially see the messages issued by *every* system in the sysplex, it can also potentially issue commands to *any* system in the sysplex.

Additionally, in a sysplex, some of the keywords specified in the CONSOLxx member have a sysplex-wide effect and are only processed by the first system to be IPLed into the sysplex. These keywords will be ignored by any subsequent systems as they IPL.

In this section, we assume that the starting position is that there is a pool of consoles in use by the target sysplex systems, and that every console can see the messages from every system in the sysplex and issue commands to every system in the sysplex. In addition, we assume that the incoming system has its own pool of consoles, and those consoles only have access to the incoming system.

When you move a system into the sysplex, there are really two levels of merging that you need to consider:

- The first one is that as soon as the incoming system joins the target sysplex, all the systems can now potentially see the console traffic from the incoming system, and correspondingly, the incoming system can see the console traffic from the other systems. While you can set up your consoles (in CONSOLxx) so that each console only appears to receive the same messages as prior to the merge, the important thing is that each console has the *capability* to see all messages.

  Equally, every console in the sysplex can potentially use the ROUTE command to send commands to any other system in the sysplex.

- The second, and secondary, level to consider is whether you will merge the *functions* of the consoles. You might decide to have one console that handles tape messages for all systems in the sysplex. Or you might split it the other way and have a console that handles all messages, but only from one or a subset of systems. This depends largely on your target environment (BronzePlex, GoldPlex, or PlatinumPlex).

In a BronzePlex configuration, you have to consider how important is it to maintain segregation between the incoming system and the existing systems in the target sysplex. You can set up your consoles in CONSOLxx so that some consoles will only see messages from the incoming system, and others will only see messages from the target sysplex systems. However, if segregation is very important, this is not sufficient, as someone could issue a CONTROL command and change the MSCOPE for a console.

You could go a step further and protect the commands with something like RACF; however, this would require operators to logon to the console and could prove cumbersome. Even if you do this, however, there is still the problem that the messages from every system get sent to the subsystem interface on every system in the sysplex, meaning that any program listening on that interface can see the message traffic from every system.

In a GoldPlex configuration, some of the consoles from the incoming system will typically be merged with those from the target sysplex. In a GoldPlex, you would normally not be as concerned with segregating the incoming system so the fact that all systems can potentially see all messages is not considered to be an issue. This configuration may be chosen to allow some level of amalgamation of operational functions (for example, tape or print consoles into the target sysplex console configuration).

In a PlatinumPlex configuration, the consoles of the incoming system will be fully merged into the target sysplex console configuration. If physical desk space is a consideration for the merge, then this configuration would be recommended as it makes full use of the existing consoles in the target sysplex. It also delivers the fully centralized operational management capability possible with a sysplex.

Several factors may affect your decision as to the level of console merging:

- How close are you to the sysplex limit of 99 consoles

  If there are already 99 consoles in the target sysplex, you cannot have any more consoles in the sysplex. Therefore, you have the choice of reducing (by consolidating) the number of consoles in the target sysplex prior to the merge, or you will have to use the existing target sysplex consoles to also support the incoming system.

- Physical security

  You may wish (or need) to limit access to consoles to one secure area. In this case, you would probably not wish to enable SMCS consoles.

- Workload profile - is the incoming system running a similar workload

  If the target environment is a PlatinumPlex, the incoming system should be managed in the same manner as all the other systems in the sysplex, so it makes sense in this case that the consoles are fully integrated.

- Consolidation of similar operational tasks

  For example, if the printing for the incoming system will be handled in the same physical area as the printing for all the other systems, then it makes sense to have a single console that is dedicated to *all* print handling.

- Availability of physical desk space

  If you are constrained for space in the operations area, the merge of the incoming system into the target sysplex provides a good opportunity to reduce the number of consoles required to manage the configuration.

As a general statement, it makes sense to consolidate onto as few consoles as are required to let the operators effectively manage the environment, while bearing in mind the considerations for avoiding single points of failure, and any business requirements to separate operations for the various systems.

There are four types of consoles that can be used to operate a z/OS environment:

1. MCS consoles

   These consoles are attached to a local, non-SNA controller - there can be up to 99 defined in a sysplex via the CONSOLxx parmlib member.

2. EMCS consoles

   These consoles are defined and activated by product interfaces such as TSO/E, SDSF, and NetView.

3. Hardware consoles

   The HMC is used for tasks such as activating an LPAR or doing a LOAD. The HMC also provides a console capability (SYSCONS); however, this should only be used during NIP (if no NIPCONS is available) or in emergencies.

4. SNA MCS

   These full-function MVS consoles that connect through VTAM rather than channel-attached non-SNA controllers. Note that SNA consoles count towards the limit of 99 consoles in a sysplex.

To understand the current console configuration for the incoming system or target sysplex, use the **DISPLAY CONSOLE** command.

If a console becomes inoperable or de-activated, and it cannot be re-activated, then console recovery attempts to switch the console attributes of the failed console to an alternate. It determines what consoles are candidates to be an alternate by using the console group member information. The console group is defined in the CNGRPxx member of Parmlib.

To display the current console group configuration of the incoming system or the target sysplex, use the **DISPLAY CNGRUP** command. Refer to Figure 19-3 for sample output from the command. When there is a console group member active, the response from the command includes all groups that are defined, along with the consoles in each group.

```
D CNGRP
IEE679I 18.57.38 CNGRP DISPLAY 258
CONSOLE GROUPS ACTIVATED FROM SYSTEM SYSA
---GROUP--- --------------------MEMBERS--------------
MASTER   00 SYSAM01  SYSAM02  SYSAM03
HCGRP    00 *SYSLOG*
```

Figure 19-4 Displaying console groups

There is no default ALTGRP for a console. The only way for a console to have an ALTGRP is if an ALTGRP is specified via the CONSOLxx member of Parmlib, or if one is assigned via the **VARY CN** command. We recommend using ALTGRP, since this will allow for the console function to switch to consoles that are on other systems within the sysplex.

19.2.1 EMCS security

Several products (TSO/E, SDSF, System Automation for OS/390) provide the capability of using EMCS consoles. With this comes the issue of security of information being visible to the EMCS consoles, such as outstanding WTORs from all systems in the sysplex. With SDSF and the use of OPERLOG, you have the ability to view outstanding WTORs from all systems in the sysplex.

SDSF offers security via the ISFPRMxx configuration member using the ACTION keyword of the ISFGRP macro. The ACTION keyword will then limit which WTORs are visible in the SYSLOG/OPERLOG for the relevant ISFGRP definition. Refer to *z/OS SDSF Operation and Customization,* SA22-7670 for further information.

The TSO/E and System Automation for OS/390 products also offer the same sort of features to secure EMCS consoles via RACF resources and profiles. Refer to the product-specific manuals for further information and customization.

19.2.2 Systems Network Architecture (SNA) MCS

SMCS consoles are MCS consoles that use VTAM services for input and output. The SMCS consoles can be real 3270-type devices, but they can also be 3270 emulators. SMCS supports VTAM LU Type 0 or Type 2.

SMCS is a VTAM application and is therefore reliant on VTAM being available for communication. SMCS consoles were first introduced with z/OS 1.1. SMCS and MCS consoles can coexist within a system or sysplex.

Figure 19-5 on page 315 shows sample CONSOLxx statements to define an SMCS console.

```
CONSOLE  DEVNUM(SMCS)
         NAME(CON1)
         ALTGRP(MASTER)
         AREA(NONE)
         AUTH(MASTER)
         CMDSYS(*)
         CON(N)
         DEL(R)
         LEVEL(ALL)
         LOGON(REQUIRED)
         MFORM(T,S,J,X)
         MSCOPE(*ALL)
         PFKTAB(PFKTAB1)
         RNUM(28)
         ROUTCODE(ALL)
         RTME(1/4)
         SEG(28)
         USE(FC)
```

Figure 19-5 Sample CONSOLxx statements for SMCS console

While SMCS consoles provide great flexibility, there are some restrictions on the SMCS consoles:

- They cannot handle Synchronous WTORs (DCCF).
- SMCS consoles are not available during NIP.
- SMCS consoles cannot be used before VTAM has started.
- SMCS consoles must be activated differently than MCS consoles (no **VARY CONSOLE** and **VARY CN,ONLINE** support).
- SMCS does not support printer consoles and SMCS consoles cannot be used as hardcopy devices.

On the other hand, SMCS consoles:

- Support all of the CONTROL (K) commands
- Can be the sysplex Master Console
- Can go through console switch processing
- Can be removed by IEARELCN
- Can be logged on to
- Can receive route codes, UD messages, and so on
- Can issue commands
- Are included in the 99-console limit
- Must be defined in the CONSOLxx member of Parmlib
- Have the same set of valid and invalid characters

19.2.3 Console recommendations

It is recommended that consoles be consolidated in the target sysplex before a merge, especially if the number of MCS and subsystem consoles in your target sysplex is already at (or close to) 99. The maximum number of consoles (MCS and subsystem) in a sysplex is 99, so adding the consoles from an incoming system into the target sysplex could potentially exceed this number.

If you have not named your MCS and subsystem consoles, we recommend that you do so now. If you have products installed that have the capability of using EMCS, we recommend that you configure them to use EMCS rather than subsystem consoles.

If you use, or plan to use, SNA MCS consoles, you must consider securing these types of consoles, as SNA MCS consoles could be logged onto from anywhere in your network.

19.2.4 Considerations for merging consoles

This section contains, in checklist format, a list of items that must be considered as you review your console configuration.

Table 19-2 Considerations for merging consoles

Consideration	Note	Type	Done
Review CONSOLxx parms that are sysplex-wide.	**1**	**G,P**	
Ensure no consoles are defined with MSCOPE=*ALL.	**2**	**B,G,P**	
Set an appropriate value for RMAX.	**3**	**B,G,P**	
Name all consoles (MCS and Subsystem).	**4**	**B,G,P**	
Carry out console consolidation on the existing environment prior to the merge.	**5**	**B,G,P**	
If at all possible, share a single CONSOLxx member between all systems in the sysplex.	**6**	**G,P**	
There is only one Master Console per sysplex.	**7**	**B,G,P**	
Route Undelivered (UD) messages to a destination other than the Master Console.	**8**	**B,G,P**	
Ensure alternate console groups are being used.	**9**	**B,G,P**	
Review automation requirements.	**10**	**B,G,P**	
Provide console configuration documentation for operations.	**11**	**B,G,P**	
Console security - requirement to logon to console.	**12**	**B,G,P**	
Review NIPCONS definitions.	**13**	**B,G,P**	
Console address for stand-alone dump.	**14**	**B,G,P**	
Determine the number of EMCS consoles in use.	**15**	**B,G,P**	
Check that routing codes are consistent across all consoles within the sysplex.	**16**	**B,G,P**	
Assign a minimum of two MCS consoles for sysplex, and remove any single points of failure.		**B,G,P**	

Consideration	Note	Type	Done
Attach the Master Console and an alternate Master Console to the fastest processors in the sysplex to ensure that there is sufficient processing power to process all messages as they are issued.		**B,G,P**	
Verify that software products are exploiting Extended MCS console support to avoid exceeding the 99 MCS console limitation.		**B,G,P**	

The “Type” specified in Table 19-2 on page 316 relates to the target sysplex environment—**B** represents a BronzePlex, **G** represents a GoldPlex, and **P** represents a PlatinumPlex. Notes indicated in Table 19-2 on page 316 are described below:

1. When operating in a multi-system sysplex, certain CONSOLxx keywords have a sysplex scope. When the first system is IPLed, the values specified on these keywords take effect for the entire sysplex. Table 19-3 summarizes the scope (system or sysplex) of each CONSOLxx keyword.

Table 19-3 Scope of CONSOLxx keywords

CONSOLxx statement	System scope	Sysplex scope
CONSOLE	DEVNUM UNIT PFKTAB	NAME ALTERNATE ALTGRP AUTH USE CON SEG DEL RNUM RTME AREA UTME MSGRT MONITOR ROUTCODE LEVEL MFORM UD MSCOPE CMDSYS SYSTEM LOGON LU
INIT	PFK MONITOR CMDDELIM MPF UEXIT MMS MLIM LOGLIM NOCCGRP APPLID	AMRF RLIM CNGRP ROUTTIME GENERIC

CONSOLxx statement	System scope	Sysplex scope
DEFAULT	LOGON HOLDMODE ROUTCODE SYNCHDEST	RMAX
HARDCOPY	All keywords	

2. We do not recommend coding MSCOPE=*ALL for any console, as this has the potential of generating large amounts of unsolicited message traffic to the corresponding console. Be aware that the *default* value is MSCOPE=*ALL, so you will to specifically override this to change it. The MSCOPE parameter allows you to specify the systems in the sysplex from which this console is to receive messages not explicitly routed to this console. We recommend that you set this value to MSCOPE=*. The asterisk (*) indicates the system to which this CONSOLE is attached.
3. The RMAX value specifies the maximum value of a reply id in the sysplex, and also determines the size of the reply id displayed in the message text. Specifying a value of 9999 will result in 4-character reply ids. Note that the value that you specify for RMAX can affect the size of the XCF Couple Data Set. Refer to *z/OS MVS Initialization and Tunning Reference,* SA22-7592 and *z/OS MVS Setting Up a Sysplex,* SA22-7625 for further information.
4. When a system is IPLed and rejoins the sysplex, a new consoleid gets allocated if the consoles on that system prior to the IPL were not named. As a result, repeated IPLs may result in exceeding the limit on the number of consoles in the sysplex. Should this happen, you can recover by using the IEARELCN program provided in SYS1.SAMPLIB. The only other way to free up those consoleids is by a sysplex-wide IPL.

   Subsystem consoles also take from the pool of 99 consoles, so you should:

   – Use EMCS rather than subsystem consoles wherever possible.
   – For products that do not support EMCS consoles, make sure you name all defined subsystem consoles.

   If you wish, you can use system symbols in consoles names—this allows you to have a single CONSOLxx member that is shared by all systems, and still have unique names on each system.
5. Wherever possible, consolidate consoles that perform duplicate functions. This should result in a simpler operations environment and free up consoleids.
6. We do not recommend using multiple CONSOLxx members. In most cases, the use of system symbols will allow system-unique values if needed. However, if there is still a requirement for separate CONSOLxx members, ensure that the sysplex-wide parameters are consistent across all members of the sysplex.
7. The sysplex only has one SYSPLEX MASTER console. The SYSPLEX MASTER console is established when the first system with consoles attached that have MASTER authority is IPLed into the sysplex.
8. Undelivered (UD) messages are, by default, sent to the SYSPLEX MASTER console. We recommend sending these messages to the HARDCOPY device instead.
9. Alternate console groups are defined in the CNGRPxx member of Parmlib. This member is used to define console groups as candidates for switch selection in the event of a console failure. You can specify both MCS and EMCS consoles as candidates. Every console should be part of an alternate console group, and should have an ALTGRP specified on the console definition in CONSOLxx.

10. Automated message handling may need to be modified to determine the originating system that issued the message before taking action.

    Also under the topic of automation, you should establish routines to automatically handle WTO/WTL buffer shortages, or use the WTO automation capability in msys for Operations.

11. Operations should have a configuration diagram showing (at a minimum) the defined consoles, which system each is attached to, which console group each console is a member of, and the alternate group specified for each console. Document console operations and recovery procedures, including the use of command prefixes, ROUTE commands, recovering from console failures, performing console switches, switching console modes, and recovering from buffer shortages.

    Refer to the IBM Redbook *Parallel Sysplex Operational Scenarios,* SG24-2079, for further information.

12. Console security may need to be considered, depending on the requirements of the incoming system. If there is a requirement to secure consoles, you will need to review the LOGON parameter of the CONSOLxx member in parmlib. Refer to *z/OS MVS Initialization and Tunning Reference,* SA22-7592 for further information.

13. At the early stages of initialization of all MVS components and subsystems, there are no MCS consoles available for operator communication. Consoles eligible for NIPCONS are defined via HCD where you specify the address of the console. If you have not defined any devices as being eligible for NIPCONS in the IODF, or if none of those are attached to the system currently being IPLed, NIP will use the HMC interface.

14. Review the stand-alone dump procedures for the incoming system. The console keyword of the AMDSADMP macro indicates the device numbers and device types of the SADMP consoles to be used by the SADMP program. As the incoming system may have a different address coded in the console keyword, and the device address may no longer be valid in the merged environment, we recommend that you specify CONSOLE=SYSC in the AMDSADMP macro. SYSC is a constant, representing the hardware system console. Refer to *z/OS MVS Diagnosis: Tools and Service Aids,* GA22-7589 for further information.

15. A large number of EMCS consoles could slow down the IPL for systems joining the target sysplex. You can use the **D EMCS,ST=L** command to display the number of EMCS consoles currently in use in the sysplex. Figure 19-6 shows a sample output from the command. If you do not know who owns a particular console (it is not always obvious from the console name), you can issue a **DISPLAY EMCS,F,CN=consname** command to find the owning address space.

    Some products can consume huge numbers of EMCS consoles depending on how they are set up, so it is wise to monitor this number and try to address any products that use a larger number than necessary.

```
D EMCS,ST=L
IEE129I 23.21.54 DISPLAY EMCS 360
DISPLAY EMCS,ST=L
NUMBER OF CONSOLES MATCHING CRITERIA: 49
SYSA     MSOSASIR AOPER5SA SYSB     MSOSBSIR AAUTO2SB SYSC
WELLIE2  *ROUTO61 AUTLOGSA AUTMSGSA AUTRPCSA ATNET1SA AUTJESSA
AUTRECSA ARONNSA  *ROUTO62 SUHOOD   *ROUTO58 AAUTO1SA *ROUTO59
AAUTO1SB *ROUTO60 ALBERTO  RONN     KYNEF    BERNICE  AWRKO2SA
AUTGSSSA ATXCF2SA AWRKO3SA AWRKO1SA AUTMONSA VALERIA  RAMJET
*OPLOGO1 AAUTO2SA *OPLOGO2 AOPER5SB *OPLOGO3 HIR      ATBASESA
ATSHUTSA AUTCONSA AHWOO1SA AUTXCFSA AUTSYSSA AHW4459  GRANT
```

Figure 19-6 Output from Display EMCS command

16. Routing codes are associated with each message. When defining a console, you can specify which routing codes are to be sent to that console. More than one routing code can be assigned to a message to send it to more than one console. The system will use these route codes and the console definitions to determine which console or consoles should receive the message.

    Route codes are not shown with the message on the console. To determine the route codes for each console in your sysplex configuration, issue the **DISPLAY CONSOLE,A** command. You can limit, and dynamically modify, the types of messages sent to a console by assigning a route code or codes to a console. You can specify the route codes on the **VARY CN** command. For example, you would use the **VARY CN(CONS01A),CONSOLE,ROUT=(1,2,9,10)** command to assign route codes 1, 2, 9, and 10 to a console named CONS01A.

In addition to the points mentioned here, you should also refer to two excellent white papers:

- "*Console Performance Hints and Tips for a Parallel Sysplex Environment*", available at:

  http://www.ibm.com/servers/eserver/zseries/library/techpapers/pdf/gm130166.pdf
- "*Parallel Sysplex Availability Checklist*", available at:

  http://www.ibm.com/servers/eserver/zseries/library/whitepapers/pdf/availchk_parsys.pdf

For general information about console use and planning, refer to:

- *z/OS MVS Routing and Descriptor Codes,* SA22-7624
- *z/OS MVS System Commands,* SA22-7627
- *z/OS MVS Planning: Operations,* SA22-7601
- *z/OS MVS Initialization and Tunning Reference,* SA22-7592

19.2.5 Console philosophy

When deciding how to configure consoles in a sysplex, the consoles should ideally be configured based on *functions,* rather than on systems. You should try to use the existing console environment already configured in the target sysplex. Another consideration is to try and use an automation focal point, rather than MVS consoles, for day-to-day interactions with the systems. Using this approach, the merge of an incoming system into the target sysplex should not result in you having any more consoles after the merge than you did before the merge.

Note: The exception to the above would be if you decided on a BronzePlex configuration, as there would be minimal configuration changes.

19.3 General operations procedures

If the operators are familiar with managing a sysplex environment and they should also be familiar with managing the incoming system, there should be no major issues subsequent to the merge. More likely, it will just take some time to get used to the changes that will have taken place as a result of the merge.

19.3.1 IPL procedures

When merging a system into a sysplex, a review of sysplex operational procedures, including IPL, recovery, removing, and adding systems, should be done to take into account any additional requirements of the incoming system or the current target sysplex.

Operations will need to understand how to correctly initialize a sysplex environment with multiple systems, and what the procedure is for removing a system(s) from the sysplex.

If your incoming system was in a GRS environment that was not in a sysplex, you need to update your operational procedures when it joins the target sysplex; the **V GRS** command is not valid in a sysplex environment.

19.3.2 Sysplex Failure Management and Automatic Restart Manager

The goals of Sysplex Failure Management (SFM) and Automatic Restart Manager (ARM) are complementary. SFM helps keep the sysplex up and running, even if one of the members of the sysplex experiences a problem, and ARM keeps specific work up and running in the sysplex, even if the system the work was running on suffers an outage.

SFM manages system-level problems and partitions systems out of the sysplex in response to those problems. ARM manages subsystems and restarts them appropriately after either subsystem-level or system-level failures occur.

Operations staff should be familiar with whether SFM and/or ARM policies are active in the incoming system or target sysplex environments.

There needs to be operational awareness if SFM and/or ARM are active, as this may affect the way you carry out the day-to-day operational tasks within the sysplex. The use of SFM and/or ARM may also affect any automation environment currently in place. We recommend that you review the impact that SFM and/or ARM may have in your automation environment accordingly.

An SFM policy can be used to assist in the operational management of a sysplex environment. An example of this could be to automate the following message that is issued when removing a system from the sysplex:

```
IXC102A XCF IS WAITING FOR SYSTEM sysname DEACTIVATION. REPLY DOWN WHEN
       MVS ON sysname HAS BEEN SYSTEM RESET.
```

Figure 19-7 Partitioning a system out of a sysplex

By using an SFM policy to act upon this message, there will be no requirement for operator intervention. The system will automatically be reset and partitioned out of the sysplex. Without SFM in place, operations must manually do a System Reset of the LPAR to release any hardware Reserves that may still be in place, followed by replying `DOWN` to the IXC102A message.

For information on how to define and activate an SFM Policy, refer to *z/OS MVS Setting Up a Sysplex,* SA22-7625.

ARM has the ability to restart failed address spaces on the same system as the failure, or to do cross-system restarts. The implementation of an ARM policy could affect the way your automation environment handles restarts of address spaces.

19.3.3 Operational Efficiency

Use the following functions to maintain operational efficiency in a sysplex environment:

- Implement message suppression using a combination of MPFLST and automation on all systems. A review of the contents of MPFLST and automation product should also be done to ensure that there is no duplication of actions against messages.

- The Command Prefix Facility allows the entering of a subsystem command on any system in the sysplex and have the command routed to whichever system the subsystem is running on.
- The CMDSYS parameter in CONSOLxx specifies which system commands entered on that console are to be automatically routed to.
- The SYNCHDEST parameter in CONSOLxx controls the handling of synchronous WTORs.
- The ROUTE command sends commands from any console to various systems in the sysplex.
- SDSF in conjunction with OPERLOG can provide a sysplex-wide view of all the syslogs and outstanding WTORs from each system. If there is a requirement to limit or restrict the ability to view such displays, then customization of the SDSF parameters is required. Further information on this can be found in *z/OS SDSF Operation and Customization,* SA22-7670.

19.3.4 JES2 Multi-Access Spool

Because it is possible to have multiple JES2 MASs in a single sysplex, there must be an awareness of how batch jobs are managed in such an environment and the possible interaction with any scheduling package. Regardless of your target environment (BronzePlex, GoldPlex, or PlatinumPlex) and whether that environment contains one or more MASs, the operators need to understand how the new environment will change the way they manage jobs in the incoming system and the systems in the target sysplex.

19.3.5 WLM-Managed Initiators

If you are running Workload Manager in goal mode, you have the ability to have WLM manage all or some of your job classes. Operational management of WLM-Managed initiators and job classes is fundamentally different for such an environment.

If the incoming system is being merged into an existing JES2 MAS environment on the target sysplex, and the target sysplex is running in WLM goal mode with WLM-Managed Initiators, WLM will start managing the initiators for the managed job classes on the incoming system as soon as it joins the sysplex. You will need to review your initiator class definitions to assess the impact of this change. For more information about WLM-Managed Initiators, refer to Chapter 4, “WLM considerations” on page 51.

19.3.6 Automatic Tape Switching (ATS Star)

Operations personnel need to be aware of the ability to share tape drives between systems in a sysplex. For environments that are z/OS 1.2 or lower, the sharing is done using the IEFAUTOS CF structure. If your environment is z/OS 1.2 with APARS OW51103 and OW50900, tape sharing (known as ATS STAR) is implemented using GRS rather than the IEFAUTOS structure to control serialization of the tape drives. It is possible to share tapes using a mixture of ATS STAR and IEFAUTOS if your sysplex consists of a mix of systems, those with the ATS STAR support and those without that support. For further information, refer to *z/OS MVS Setting Up a Sysplex,* SA22-7625

19.3.7 Shared sysres

The ability to share the system residence volume (sysres) between multiple systems (and the impact of doing so) needs to be made known to operations personnel. By sharing the sysres, you ensure that all the systems running off that sysres are using identical release and service levels of the products on the sysres. When sharing the sysres, any change that is made to it will affect *all* the systems that are using that sysres. Operations will need to ensure that the HMC LOAD Profile or grouping is configured correctly to manage a shared sysres environment.

Consideration does need to be given to how many systems in the sysplex will IPL from the same sysres, as you may want to minimize the potential single point of failure to a subset of systems. You could logically separate your target sysplex environment into a “production” environment, a “development” environment, and a “sysprog” environment, and have each of these IPL from a different sysres. Any failure of the sysres within each of these sub-environments would then only impact the systems belonging to that environment. The subject of shared sysres is discussed in more detail in 10.3, “Sharing sysres” on page 147.

19.3.8 Useful Commands

The following MVS commands can be used to provide useful information on the configuration of the systems within the target sysplex or incoming system:

- **D SYMBOLS** - display system symbols currently in effect for the system the command is issued on.
- **D IPLINFO** - display information about the date and time of the IPL, the release level of the system, the IEASYSxx and IEASYMxx members that were used at IPL time, the LOADxx member used for the IPL, the IODF device number, and the IPL device number and volume serial.
- **D U,IPLVOL** - display the device number from which the system was IPLed.
- **D EMCS,ST=L** - display the names of the EMCS consoles.
- **D C** - display the current console configuration.
- **D CNGRP** - display the console group definitions.

Refer to *Parallel Sysplex Operational Scenarios,* SG24-2079 for further information.

19.3.9 Housekeeping

When merging a system into the sysplex, certain housekeeping jobs may need to be updated to include the incoming system. If System Logger has been implemented for OPERLOG and/or EREP, a review of the housekeeping jobs is required to ensure that the data generated in these environments is processed and maintained accordingly. The IEAMDBLG and IFBEREP members of SYS1.SAMPLIB provide samples of how to define housekeeping jobs to manage an OPERLOG and/or EREP environment that has been configured to use the System Logger.

Other housekeeping functions, such as SMF dump processing, will need to be reviewed once the systems have been merged, as there could be a common SMF switching exit (IEFU29) being used which may cause conflict with the target sysplex.

19.4 Tools and documentation

In this section we provide information about tools and documentation that may be of assistance to you.

19.4.1 Tools

The following tools may be of assistance for this merge task:

TSO Operator Presentation Sample (TOPS)

The TSO/E Operator Presentation Sample (TOPS) is an ISPF dialog written in REXX. TOPS presents a convenient way to use the TSO/E Console services and provides a useful dialog for z/OS operations from TSO/E. SYS1.SAMPLIB(IEATOPSD) contains detailed information on how to install this sample package.

IEECMDPF

SYS1.SAMPLIB(IEECMDPF) is a sample program to define a Command Prefix that is equal to the system name.

IFBEREPS

SYS1.SAMPLIB(IFBEREPS) is sample JCL to use the LOGREC logstream subsystem data set interface.

IFBLSJCL

SYS1.SAMPLIB(IFBLSJCL) is sample JCL to define the LOGREC log stream.

IEARELCN

SYS1.SAMPLIB(IEARELCN) is a sample program that can remove a console definition from a single system or sysplex.

IEAMDBLG

SYS1.SAMPLIB(IEAMDBLG) is a sample program to read records from the OPERLOG log stream and convert them to syslog format. It can also be used to delete records from the log stream.

IEACONXX

SYS1.SAMPLIB(IEACONXX) is a sample CONSOLxx member that utilizes console and subsystem console naming and also has definitions for SNA MCS consoles.

19.4.2 Documentation

The following documentation may be of assistance for this merge task:

- An IBM White paper entitled “Console Performance Hints and Tips for a Parallel Sysplex Environment”, available on the Web at:

  `http://www.ibm.com/servers/eserver/zseries/library/techpapers/pdf/gm130166.pdf`
- *Controlling S/390 Processors Using the HMC,* SG24-4832
- *Hardware Management Console Operations Guide,* SC28-6809
- *OS/390 MVS Multisystem Consoles Implementing Sysplex Operations,* SG24-4626
- *zSeries 900 System Overview,* SA22-1027

- *zSeries 900 Technical Guide,* SG24-5975
- *z/OS MVS Planning: Operations,* SA22-7601

20

Hardware configuration considerations

This chapter discusses the considerations for hardware configuration, management, and definition when you are merging a system into an existing sysplex. In this chapter we will discuss hardware configuration items to be considered when merging a system into a sysplex, including the following:

- A migration table describing what is suitable for BronzePlex, GoldPlex or PlatinumPlex configurations.
- A checklist of items to consider when merging.
- A methodology for carrying out the merge.
- A list of tools and documentation that can assist with these tasks.

20.1 Introduction

The considerations discussed in this chapter will be described against the three following target environments:

- The BronzePlex environment is where the minimum number of resources are shared; basically only the DASD volumes where the sysplex CDSs reside. A BronzePlex configuration would typically consist of the minimal amount of sharing necessary to qualify for PSLC or WLC license charging.
- The GoldPlex environment is where a subset of the configuration is shared. This might consist of the DASD volumes where the sysplex CDSs are located and some or all of the system volumes.
- The PlatinumPlex environment is where all resources within the sysplex configuration are shared. This configuration potentially provides the largest benefits in terms of flexibility, availability, and manageability.

20.2 Assumptions

The following assumptions have been made in this chapter to establish a baseline configuration for the environment:

- All systems and attached peripheral devices are currently located in the same machine room.
- There are two CFs.
- The incoming system will remain in the same LPAR that it is in at the moment.
- All CPCs containing members of the sysplex are accessible from all the HMCs in the HMCplex.
- There is currently one IODF for all the systems in the target sysplex, and another separate IODF for the incoming system.

20.3 Considerations when merging systems into a sysplex

When merging a system into the target sysplex, there are a number of physical considerations that must be taken into account. To assist you in understanding these requirements, the items in Table 20-1 need to be reviewed bearing in mind your target configuration.

Table 20-1 Physical considerations when merging systems into a sysplex

Consideration	Note	Type	Done?
Consider how many IODFs you plan to use - a single IODF for the sysplex, or multiple IODFs.	**1**	**B,G,P**	
Check for duplicate device numbers.	**2**	**G,P**	
Review NIP console requirements.	**3**	**G,P**	
Review SQA storage considerations	**4**	**B,G,P**	
Review/update Channel Measurement Block (CMB) parameter of IEASYSxx in parmlib	**5**	**B,G,P**	
Review Hardware System Area (HSA) and expansion factor.	**4**	**B,G,P**	

Consideration	Note	Type	Done?
Investigate consolidating into a single OS configuration if possible.	**7**	**B,G,P**	
Review use of Esoterics.	**8**	**G,P**	
Investigate effect of multiple IODFs on the ability to do a sysplex-wide dynamic reconfiguration from the HCD dialogs.	**9**	**B,G,P**	
Review CF connectivity.	**10**	**G,P**	
Create new CFRM policy and ensure all systems have connectivity to all CDS DASD.	**11**	**G,P**	
Update your XCF signalling configuration to support the incoming system.	**12**	**B,G,P**	
Ensure the incoming system has the required Sysplex Timer connectivity, and update the CLOCKxx member accordingly.	**13**	**G,P**	
Check that the required CFLevels are available.	**14**	**G,P**	
Review ESCON Logical Path requirements.	**15**	**G,P**	
Review HMC configuration.	**16**	**B,G,P**	
Check Hardware & software microcode levels.	**17**	**B,G,P**	

The “Type” specified in Table 20-1 on page 328 relates to the sysplex target environment—**B** represents a BronzePlex, **G** represents a GoldPlex, and **P** represents a PlatinumPlex. Notes indicated in Table 20-1 on page 328 are described below:

1. Depending on your configuration, we recommend that a single IODF containing all of your I/O definitions be used wherever possible. This will greatly reduce the amount of work required to maintain your I/O configuration definitions. If you want to limit certain systems to a subset of devices, you can still use a single IODF, but in this case use multiple operating system configurations within the IODF and specify the appropriate config ID within the LOADxx member.

   If you feel that your I/O configuration is too large for effective management within a single IODF, you may consider having a master IODF that you can carry out your changes to, and then generate a subset IODF that is transferred to the corresponding target system where it is imported and used as the IODF for that system.

   The subset IODF constitutes a fully functional IODF. When it is built from a master IODF, the processor tokens are preserved. There are no strict rules about what a subset IODF must consist of, however, it typically contains:

   – A processor configuration with its related OS configuration, or
   – All I/O configurations describing the LPARs in a single CPC, or
   – All I/O configurations describing the systems of a sysplex

   The contents of a subset IODF are specified in a configuration package. Refer to *z/OS Hardware Configuration Definition (HCD) Users Guide*, SC33-7988 for information on the Master IODF concept, subset IODFs, and working with configuration packages.

   This subject is discussed in more detail in 20.4, “Single or multiple production IODFs” on page 333.

2. Merging the incoming system into the target sysplex could present duplicate device numbers that will need to be resolved. To check for this, use the HCD panels to get a printout out of both configurations, checking that any duplicate device numbers are actually referring to the same physical device. Any duplicates that are *not* referring to the same device will need to be resolved. As the merge of the system will require an outage, it

is an opportune time to resolve any duplicate device numbers. See 20.8, “Duplicate device numbers” on page 341 for a further discussion about this.

3. It is possible, in a large sysplex, that not every system will have a channel-attached console in a secure area. In this situation, you can use the HMC as the NIPCONS for those systems. A advantage of using the HMC is that you can scroll back and see messages issued earlier in the NIP process, whereas with a channel-attached NIP device, once the messages roll off the screen, they can’t be viewed until the system is fully up and running.

   On the other hand, the HMC is slower than a normal console, so if you have many messages issued during NIP, the use of the HMC may slow down the process somewhat. Refer to 19.2, “Consoles” on page 311 for further information. This is also discussed in 20.9, “NIP console requirements” on page 341

4. Virtual Storage Constraint Relief can be achieved by specifying that the UCBs for devices reside above the 16 MB line by using the LOCANY parameter in HCD. You will need to carefully review which devices are specified with the LOCANY parameter, as you may be using products that do not support this. Refer to *z/OS Hardware Configuration Definition (HCD) Users Guide*, SC33-7988 for further information.

   You may also need to review the SQA parameter of the IEASYSxx member of Parmlib. OS/390 and z/OS obtain an initial amount of SQA during IPL/NIP, prior to processing the SQA=(x,y) parameter in IEASYSxx. This initial SQA allocation may not be sufficient if a large number of duplicate volsers must be handled during NIP processing. The INITSQA=(a,b) parameter in LOADxx can be used to circumvent SQA shortages during NIP.

5. The Channel Measurement Block (CMB) accumulates utilization data for I/O devices. You need to specify, on the CMB parameter in the IEASYSxx member of Parmlib, the sum of the number of I/O devices that you need to measure that are not DASD or tape, plus the number of devices that you plan to dynamically add and measure. Refer to *z/OS MVS Initialization and Tunning Reference,* SA22-7592 for further information.

6. If you expect to make dynamic I/O configuration changes, you *must* specify a percentage expansion for HSA. The amount of HSA that is required by the channel subsystem depends on the configuration definition, processor, and microcode levels. This expansion value can only be changed at Power-On Reset (POR). The **D IOS,CONFIG(HSA)** command will display the amount of HSA available to perform dynamic configuration changes. Refer to Figure 22-1.

```
D IOS,CONFIG(HSA)
IOS506I 16.21.08 I/O CONFIG DATA 361
HARDWARE SYSTEM AREA AVAILABLE FOR CONFIGURATION CHANGES
       829 PHYSICAL CONTROL UNITS
      4314 SUBCHANNELS FOR SHARED CHANNEL PATHS
      1082 SUBCHANNELS FOR UNSHARED CHANNEL PATHS
       623 LOGICAL CONTROL UNITS FOR SHARED CHANNEL PATHS
       156 LOGICAL CONTROL UNITS FOR UNSHARED CHANNEL PATHS
```

*Figure 20-1 Output from the **D IOS,CONFIG(HSA)** command*

The “Percent Expansion” value specified on the HMC represents the portion of the HSA that is reserved for the dynamic activation of a new hardware I/O configuration. It is based on the amount of HSA that will be used to accommodate the I/O configuration in the IOCDS that is to be the target of the next Power-On Reset (POR).

The Reset Profile will need to be updated if the value for the expansion factor needs to be changed. Refer to *2064 zSeries 900 Support Element Operations Guide*, SC28-6811 for further information.

7. As part of the planning for the merge of the incoming system, you should have identified all the devices currently accessed by the target sysplex systems that the incoming system will require access to. Similarly, you should have identified which devices currently used by the incoming system will be required by the other systems in the target sysplex. This information will be part of the input into your decision about how many OS configurations you will have in the target environment.

   If your target environment is a BronzePlex, you would more than likely wish to have multiple OS configurations, especially if all systems are using the same master IODF. The use of multiple OS configurations can help you ensure that each system in the sysplex only has access to the appropriate subset of devices. Relying on varying certain devices offline at IPL time is not as secure as not making them accessible to the operating system in the first place.

   If your target is a GoldPlex, the decision of how many OS configurations to have will depend on the degree of separation you want between the systems. If you would like the ability to bring certain devices from the “other” subplex online at certain times, it may be better to have just one OS configuration. On the other hand, if your requirement is closer to a BronzePlex and you definitely do not want either subplex to be able to access the other subplex’s devices, then two OS configurations would be more appropriate to that requirement.

   If your target configuration is a PlatinumPlex, we recommend that you create a single, merged OS configuration. By having just a single OS config, you reduce the complexity and potential errors introduced when having to maintain multiple OS configurations for a multisystem sysplex environment. Additionally, you ensure that the systems will behave consistently (for example, specifying UNIT=TAPE will refer to the same pool of devices regardless of which system the job runs on).

   This topic is discussed further in 20.11, “Single or multiple OS Configs” on page 342.

8. When deciding to merge a system into an existing sysplex, you will need to consider the possibility of conflicting or no-longer-valid esoterics.

   If your target environment is a BronzePlex, you will more than likely have a different OS configuration for each subplex. In this case, you can probably leave the esoteric definitions unchanged.

   If you are moving to a single OS configuration, you will need to ensure that the esoterics effectively support all systems. You should check the existing definitions in each system, addressing any discrepancies, and bearing in mind that in a GoldPlex all of the devices will not necessarily be online to all systems.

   The decision to merge an incoming system into the target sysplex provides an opportunity to clean up esoterics that may no longer be required in the target sysplex environment. However, before you proceed with any deletes, you must ensure that esoterics were never used when cataloging data sets. This restriction is documented in the section that discusses defining non-VSAM data sets in *z/OS DFSMS Access Methods Services for Catalogs,* SC26-7394.

   This topic is discussed further in 20.10, “Eligible Device Table and esoterics” on page 341.

9. If you decide to configure your environment to have multiple IODFs, you will lose the ability to initiate a single sysplex-wide dynamic I/O reconfiguration activate from the HCD panels unless all the IODFs have duplicate names. The reason for this is that the HCD dialogs checks the high level qualifier of the current IODF on every system against the high level qualifier of the IODF that is being activated, and refuses the activate if they are not the same. Even if the HLQs were the same, the IODF suffix must also be the same, because

the same suffix is used on the ACTIVATE command that is issued on every system—there is no capability to specify different suffixes for the different systems.

10. Because the incoming system was not in the sysplex prior to the merge, you will need to add connectivity from that LPAR to all the CFs in use in the sysplex (this presumes there are already other LPARs on the same CPC already communicating with the CFs in question). Generally speaking, two CF links from each CPC to each CF should provide acceptable performance. To verify this, check that the utilization of each link is below about 30%. If utilization increases above that amount, you should consider adding more CF link capacity, or using faster CF link technology.

11. There can only be one CFRM Policy active for the entire sysplex. If you are merging a system that was already connected to an existing CF, you will need to review its CFRM Policy definitions so that they can be migrated to the CFRM Policy of the target sysplex. This is discussed in more detail in 2.4.1, “Considerations for merging CFRM” on page 28.

    The DASD volume containing the Primary, Alternate, and Spare CFRM CDSs must be accessible to all systems in the sysplex.

12. As soon as the incoming system joins the target sysplex, it will require XCF connectivity to all the other systems in the sysplex. This connectivity can be obtained using CTCs (ESCON or FICON), CF signalling structures, or a combination of the two.

    The great advantage of using only CF structures is that no additional CTC paths are required as you add more systems to the sysplex. The number of CTC paths you need to provide full connectivity with redundancy is n*(n-1), where n is the number of systems in the sysplex. So, you can see that as the number of systems in the sysplex grows, each additional system requires a significant number of additional CTC paths, along with all the attendant work to define and maintain this information in HCD and in the COUPLExx members. On the other hand, if you are only using CF structures for XCF signalling, the only change required when you add a system to the sysplex should be to increase the size of the signalling structures. To assist in the resizing of the XCF Structure, you can use the Parallel Sysplex Coupling Facility Structure Sizer (CFSizer), available on the Web at:

    `http://www.ibm.com/servers/eserver/zseries/cfsizer/`

    If you will only be using ESCON CTCs for your XCF signalling, you will need to update the PATHIN/PATHOUT definitions defined in the COUPLExx member of Parmlib, to add the new paths between the incoming system and the other systems in the target sysplex. If you have not established a numbering scheme for your ESCON CTCs, this may be an opportune time to do so. For a large CTC configuration, it can be difficult and complex to design and define the CTC connections. To help minimize the complexity involved in defining such a configuration, IBM developed a CTC device numbering scheme. The IBM CTC device numbering scheme is discussed in detail in the *zSeries 900 ESCON Channel-to-Channel Reference*, SB10-7034.

    If your target sysplex configuration will be utilizing FICON, we recommend that you use FICON for your CTC communications. Unlike the ESCON channel CTC communication, which uses a pair of ESCON CTC-CNC channels, the FICON (FC mode) channel CTC communication does not require a pair of channels because it can communicate with any FICON (FC mode) channel that has a corresponding FCTC control unit defined. This means that FICON CTC communications can be provided using only a single FICON (FC mode) channel per processor. Further information on FICON CTC implementation can be found in the IBM RedPaper entitled *FICON CTC Implementation*.

13. When merging the incoming system into the target sysplex, you will need to ensure that the incoming system has access to the same Sysplex Timers as the other systems in the sysplex. If there are other LPARs on the same CPC as the incoming system that are already part of the target sysplex, then you know that the required connectivity already exists.

If the incoming system is the first LPAR on its CPC to join the target sysplex, the required connectivity may not exist and must be put in place before the merge. Note that it is not sufficient to just have connectivity to *a* Sysplex Timer—it must be the *same* Sysplex Timer that the other members of the sysplex are connected to.

Finally, remember that the CLOCKxx member of Parmlib must be updated to indicate that the incoming system will be using an external time reference (by specifying ETRMODE YES).

14. You will need to ensure that the Coupling Facility Control Code (CFCC) level that is running in your CF is at a high enough level to support any functions that may be in use on the incoming system prior to the merge. A single CPC cannot support multiple CF levels. This is discussed further in 2.4, "Coupling Facility Resource Management" on page 27. Also, the following Web site contains a "Coupling Facility Level (CFLEVEL) Considerations" document:

    http://www.ibm.com/servers/eserver/zseries/pso/

15. In a GoldPlex, and certainly in a PlatinumPlex, both the incoming system and the systems in the target sysplex will be accessing more DASD than they were previously. Or, to look at it another way, the DASD belonging to those systems will have more systems using them, which means more logical paths will be in use.

    Unfortunately, there is no easy way to find out how many logical paths are in use for each LCU today. The only way to get this information is to run a DSF ANALYZE report. This is discussed in more detail in 20.12, "ESCON logical paths" on page 342. For now, it suffices to say that you need to determine how many logical paths are in use for each LCU prior to the merge, and then calculate how many additional paths will be required for each LCU when you merge the incoming system and the target sysplex systems into a single sysplex.

    Even if your target environment is a BronzePlex, at a minimum the LCUs containing the sysplex CDSs will see an increase in the number of logical paths in use. So, regardless of whether you are aiming for a BronzePlex, a GoldPlex, or a PlatinumPlex, this task must be carried out. The only difference is that in a BronzePlex, there will probably be fewer LCUs affected by the merge.

16. To maintain a single point of control and ease of operational management, you should connect all of the CPCs so that they are managed from a single HMCplex. Refer to 19.1, "One or more HMCplexes" on page 302 for further discussion of this item.

17. It is important that you review the hardware microcode levels and associated software levels or fixes. If the incoming system will be remaining in the same LPAR it is in today, there should not be any concerns about CPC microcode levels (because none of that should be changing as part of the merge). However, it is possible that the incoming system will be accessing control units that it was not using previously. Similarly, the target sysplex systems could be accessing control units that were previously only used by the incoming system. In either case, you must ensure that you have any software fixes that might be associated with the microcode levels of the control units. Your hardware engineer should be able to provide you with information about the microcode levels of all hardware, and any software pre-requisites or co-requisites. You should also review the relevant Preventative Service Planning (PSP) buckets for that hardware.

20.4 Single or multiple production IODFs

There are really two questions in relation to how you manage your IODFs:

- Will you have a single IODF that contains all your hardware and software definitions, or will you have a number of IODFs, with each one containing just a subset of the configuration?

- Will all the systems use the same physical IODF data set when they IPL, or will you create copies of that data set, so that some systems use one copy, and other systems use a different one?

The decision about whether to have a single IODF containing all CPC and I/O definitions, or to have multiple IODFs, each containing a subset of the configuration, depends to some extent on the environment of your incoming system and target sysplex, and on what type of configuration you are aiming for (BronzePlex, GoldPlex, or PlatinumPlex). There are two terms we use to describe the different types of IODFs:

Master IODF A master IODF is one that contains configuration information that is not contained in any of the other IODFs. For example, if you have two CPCs, and each CPC has an IODF that only contains the information about that CPC, plus all the attached I/O devices, then you would have two master IODFs.

Cloned IODF A cloned IODF is one that is either a complete copy, or a subset, of a master IODF. The clone might be created using the COPYIODF facility or by doing an Export and Import from a master IODF.

We will first explain why it is advantageous (or even necessary) to keep all the I/O definitions in a single master IODF. After that, we discuss the considerations relating to how many cloned IODFs you should have.

As a background to this discussion, let us consider a typical modern configuration. If the configuration is small (one sysplex, one or maybe two CPCs), it is likely that all definitions will be kept in a single master IODF, so we will specifically address a larger configuration. In this case, there will probably be a number of sysplexes (production, development, and sysprog/test), a number of CPCs, and a significant number of large capacity, high performance DASD and tape subsystems. In this case, it is likely that each CPC will contain multiple LPARs, with those LPARs being spread across a number of sysplexes. Similarly, because of the capacity of the tape and DASD subsystems, and the wish to avoid single points of failure, each DASD and tape subsystem will probably contain volumes that are accessed by more than one CPC, and some of the volumes will be used by one sysplex, and other volumes will belong to a different sysplex. So, your configuration is effectively a matrix, with everything connected to everything else.

In a large, complex, environment like this, you want to do as much as possible to minimize the overhead of maintaining the configuration (by eliminating duplication where possible), you want to minimize the opportunities for mistakes (again by eliminating duplication), and you want to centralize control and management as much as possible.

Against this background, we now discuss the considerations for how many master and cloned IODFs you should have.

20.4.1 How many master IODFs

In this section we discuss the considerations that affect your decision about how many master IODFs are appropriate for your environment.

Shared control units and devices

If control units and devices are to be shared by a number of LPARs, the I/O definitions for these LPARs (and hence, the associated CPCs) should be kept in the same IODF to keep the change effort to a minimum and to avoid conflicting definitions. If you were to have one master IODF for each CPC, for example, you would need to add the definitions for the I/O devices to each master IODF rather than just defining them once.

Processor and related OS Configuration(s)

In order to fully support dynamic I/O reconfiguration, both the OS configuration that is used at IPL time (as specified in LOADxx), and the CPC, must be defined in the same production IODF. In addition, that IODF must be the one that was used to build the IOCDS that was used for the last Power-On-Reset (POR) of the CPC.

Note: To be specific, the IODF used at IPL time must either be the same one that was used to build the IOCDS, *or* it must be a clone of the data set that was created using the HCD COPYIODF or Export/Import facilities.

CF support

When connecting a CPC to a CF in HCD, both the CF and the CPC need to be defined in the IODF. If you are connecting multiple CPCs to the same CF, you either have to define the same CF over and over again in multiple master IODFs (if you have an IODF for each CPC), or else you maintain a single master IODF containing all the CPCs and the CFs they will be connecting to.

Switch connections

We recommend that you maintain your switch configurations in the same master IODF as the hardware and software configuration definitions to provide complete validation of the data path. In order to look up the port connections of an ESCON director, all connected objects to the ports of a switch have to be defined in the same IODF.

CTC checking

HCD offers you the possibility to view and verify your CTC connections that are defined through a switch. This is a very valuable function, as complex CTC configurations can be difficult to manage and debug. However, in order to be able to fully utilize this capability, all CPCs connected by the CTCs must be defined in the same IODF.

Use of HCM

If you use Hardware Configuration Manager (HCM), or are contemplating using this product, it will be necessary to use a single master IODF if you wish to have a single HCM configuration covering your total configuration. HCM does not provide the ability to extract from multiple different IODFs into a single HCM configuration.

Summary

Based on the preceding points, we recommend that you keep all your configuration information in a single master IODF. By using a single master IODF, you reduce the amount of effort and time required to implement any I/O configuration change within your sysplex.

20.4.2 How many cloned IODFs

If you *do* decide to go with a single master IODF, as we recommend, the next question is whether all systems in your environment will use the same IODF data set, or if you will create multiple copies of the IODF. The considerations in relation to this are discussed in the following sections.

HCD focal point

HCD provides a facility (option 2.11 in the HCD dialog) for centrally managing the IPL address and load parameters for CPC clusters (9672 and zSeries CPCs). To effectively manage this information for a number of CPCs, the related CPC configurations need to be kept in the same IODF (from the HCD panel you can only manage CPCs that are defined in the IODF currently specified on the I/O Definition File line on the HCD Primary panel).

If you create cloned IODFs that only contain a subset of the total configuration, you should use the focal-point function from an IODF that contains the total configuration.

Note that the information you enter in the dialog (IPL address and so on) is stored in the CPC SE, not in the IODF.

More information about this facility is available in the section entitled “Managing IOCDSs and IPL attributes across a sysplex” in *z/OS Hardware Configuration Definition (HCD) Planning,* GA22-7525.

Dynamic sysplex-wide I/O reconfiguration

HCD also provides the ability to drive a dynamic I/O reconfiguration across all the systems in the sysplex (option 2.7 in the HCD dialog). However, in order for this to be effective, all the CPCs in the sysplex, together with the OS configurations used by the sysplex systems, must be defined in the IODF currently specified on the I/O Definition File line on the HCD Primary panel.

In addition, to take advantage of the HCD sysplex-wide dynamic I/O reconfiguration feature, all the systems in the sysplex must be sharing a single IODF (or, they must all have access to two (or more) identical IODFs with the same high level qualifiers *and* the same suffix).

Summary

While HCD provides the ability to create an IODF containing just a subset of the master IODF, there is probably not a significant benefit in using this capability. All management of the contents of the IODF should always be done on the master, meaning that the subset IODFs are really only used for IPL, in which case there is not a significant difference as to whether the IODF contains the total configuration or just a subset.

If all the cloned IODFs contain the complete configuration, there is little difference from a functionality point of view whether all the systems use the same IODF clone or if there is more than one clone, within the restriction that for sysplex-wide dynamic I/O reconfiguration to work, all the cloned IODF data sets must have the same names. Given that duplicate data set names are not recommended if they can be avoided, you should ideally have a single IODF data set that every member of the sysplex uses.

The exception is in a BronzePlex, where there is minimal sharing of DASD. Because the SYS1.PARMLIB data set must be on the IODF volume or on the sysres, and you will probably not be sharing SYS1.PARMLIB in a BronzePlex, that infers that you would also not be sharing the IODF data set between the incoming system and the other systems in the target sysplex. In this case, you should use the HCD Export/Import facility to send the IODF to the incoming system and keep the same data set name on that system.

The IBM Redbook *MVS/ESA HCD and Dynamic I/O Reconfiguration Primer,* SG24-4037, contains an excellent description of dynamic I/O reconfiguration and how it relates to IODFs. It is recommended reading for anyone deciding how to best manage your IODFs.

20.5 HCD and I/O Operations

As of OS/390 1.3, the ESCON Manager (ESCM) product has been incorporated into the System Automation for OS/390 (SA/390) product family and is now called I/O Operations. I/O Operations is used to maintain and manage the dynamic switch configurations in ESCON and/or FICON switches (directors). The main functions of I/O Operations are:

- Provide operator interface-director configuration management
- Communication with directors using the channel subsystem (switches also have a PC console local end-user interface)
- Serialization of configuration changes across multiple systems sharing access through a director
- Backout of changes when any participating system detects an inability to implement the requested change
- Provide an ISPF dialog interface to enable maintenance of the switch configuration
- Provide a peer-to-peer programming interface to other hardware management components, such as HCD

20.5.1 HCD and I/O Operations relationship

HCD and I/O Operations work together to maintain the hardware configuration of the system. They each have specific roles.

HCD is used to check the definition of switch port usage and connection information for all devices and channels connected by each director. HCD checks that:

- Port connections are not duplicated.
- Devices are specified through a valid switch.
- Channel connections do not conflict.
- The path is complete.

I/O Operations is responsible for the integrity of switch changes:

- Checks for final path removal on attached systems.
- Seeks concurrence from all hosts before making a switch change.
- Retrieves switch port matrix information for manipulation and change implementation.
- Writes changes back to the switches.
- Instructs switches to change the port matrix status.
- Backs out changes, should an unrecoverable error occur.

For a target sysplex environment that may have a large I/O configuration that utilizes switches or manipulates the I/O environment regularly, using I/O Operations will greatly assist in reducing the complexity of such a task.

20.6 Activating a new I/O configuration

For a dynamic I/O reconfiguration hardware change to be successful, the actual I/O configuration in the CPC must be matched by a matching I/O configuration definition in the active IODF. A token value is used to determine whether these two I/O configurations are the same. The token value is stored in the HSA and a token value is stored in a production IODF. If these two token values are the same, then the software and hardware are said to be in synchronization (in sync).

To determine the active IODF, the OS Configuration ID, the EDT ID and the hardware processor token, issue the **D IOS,CONFIG** command. The output from this command is shown in Example 20-1.

Example 20-1 D IOS,CONFIG command output

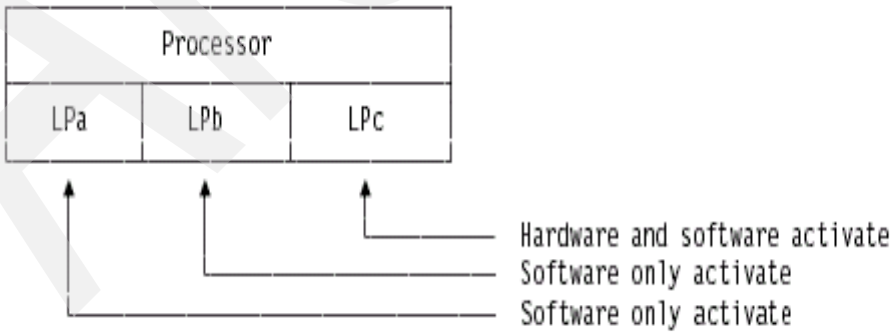
```
IOS506I 00.12.24 I/O CONFIG DATA 753
 ACTIVE IODF DATA SET = SYS6.IODF48
 CONFIGURATION ID = TRAINER        EDT ID = 01
 TOKEN:  PROCESSOR DATE     TIME     DESCRIPTION
  SOURCE: SCZP802  02-04-24 22:21:02 SYS6     IODF48
```

20.6.1 Performing the dynamic I/O reconfiguration

Once you have ensured that the hardware and software definitions are in sync, you can activate a new configuration using the production IODF representing the new configuration.

An LPAR environment requires an activation step in each system image in the CPC, regardless of which sysplex they are in (remember that the HCD-provided sysplex-wide activate works at a sysplex level rather than at a CPC level, and therefore cannot be used to do a dynamic I/O reconfig on all LPARs on a single CPC). To do this, perform a test activate followed by a software-only activate on all but the last LPAR. The test activate will inform you of any problems that might stop the activate from completing successfully, and the software-only activate implements the configuration change within the operating systems, but does not change the underlying hardware.

In the last LPAR, perform a test activate followed by a hardware/software activate in that LPAR. This is illustrated in the following diagram:



The above diagram presumes that there are three LPARs. Therefore, the activates are a software-only change in the first LPAR, a software-only change in the second LPAR, and a hardware/software change in the last LPAR.

The activation of the dynamic change can be achieved by the MVS **ACTIVATE** command or via HCD. For further information on this, refer to *z/OS Hardware Configuration Definition (HCD) Users Guide*, SC33-7988 and *z/OS MVS System Commands,* SA22-7627.

20.6.2 Dynamic I/O reconfiguration with more than one sysplex

If your final configuration will consist of more than one sysplex on the same CPC, and you are using a separate Production IODF for each sysplex, then you will need to take these items into consideration:

1. There can only be one IOCDS used for Power-On Reset (POR) for the CPC.
2. The contents of the IOCDS will have been built from one production IODF on one of the systems within your sysplexes.
3. The token stored in the HSA must match your IODF token in order for dynamic I/O reconfiguration to be allowed.
4. You should be doing all your I/O changes against a master IODF on one of your systems. From this master IODF, you will then generate your IOCDS as well as generating a clone of the master IODF to be used by the systems in the second sysplex. By using the HCD Export/Import process, you ensure that the token stored in the IODF and the token generated into the IOCDS are maintained correctly.

   When using the HCD Export/Import function, you will need to ensure that the imported IODF is cataloged correctly to the relevant systems.
5. To dynamically activate this change, you would do the following:
   - Software-only test activate (**ACTIVATE IODF=yy,SOFT,TEST**) on all systems across the sysplexes.
   - Hardware and software test activate (**ACTIVATE IODF=yy,TEST**) on all systems across the sysplexes.
   - Software-only activate (**ACTIVATE IODF=yy,SOFT**) on all but one of the systems on the CPC whose configuration is being changed.
   - Hardware and software activate (**ACTIVATE IODF=yy**) on the last system on the CPC whose configuration is being changed. You may need to specify the **FORCE** option on the **ACTIVATE** command if hardware deletes are involved.

The process outlined above is a general high-level overview. Detailed information on how to implement a dynamic I/O reconfiguration change can be found in *Hardware Configuration Definition (HCD) Planning,* GA22-7525, and *Hardware Configuration Definition (HCD) Scenarios,* SC33-7987.

20.6.3 CONFIGxx Parmlib member

After dynamic changes have been made to a system in the target sysplex, it is recommended to update the corresponding CONFIGxx member to reflect these changes.

HCD provides a function to build a CONFIGxx member containing CHP and DEVICE statements to match the environment as defined in the IODF.

By generating an up-to-date CONFIGxx parmlib member, you will be able to determine any deviations from the expected configuration by issuing the **D M=CONFIG(xx)** command on the systems in the sysplex.

Whether your target sysplex is a BronzePlex, GoldPlex, or PlatinumPlex configuration, we recommend that you go through this process after every dynamic change.

20.7 IOCDS management

There can only be one Input/Output Configuration Data Set (IOCDS) used for the Power-On Reset (POR) of a CPC. The contents of a production IODF need to have been written out to an IOCDS for the next POR of the CPC. If you have been using multiple IODFs, there is the possibility of not knowing which IODF was used to write the IOCDS to be used for the next POR.

By using HCD as a focal point and using a single IODF, you can effectively manage your I/O configuration from a single point. Using the HCD panels, you can use a production IODF to create an IOCDS on your CPC(s). You also have the ability to select which IOCDS will be used for the next POR of the CPC. The D IOS,CONFIG console command can be used to display information about the date and time the current IOCDS was created, and the name of the IODF that it was created from.

You can use HCD to verify that the production IODF that you are currently using on a system is in sync for a dynamic I/O reconfiguration change. This can be done using HCD option 2.5, as shown in Example 20-2. Note that you have to log on to each system and issue this from HCD in each system - there is no way to get a sysplex-wide view with a single command.

Example 20-2 HCD Option 2.5

```
Esssssssssssssssssssssss View Active Configuration sssssssssssssssssssssN
e                                                                        e
e Currently active IODF  . . : IODF.IODF48                               e
e    Creation date . . . . . : 02-04-24                                  e
e    Volume serial number  . : #@$#M1                                    e
e                                                                        e
e Configuration ID . . . . . : TRAINER      Trainer/GDPS Systems         e
e EDT ID . . . . . . . . . . : 01                                        e
e                                                                        e
e HSA token  . . . . . . . . : SCZP802  02-04-24 22:21:02 SYS6     IODF48     e
e                                                                        e
e Activation scope:                                                      e
e Hardware changes allowed . : Yes                                       e
e Software changes allowed . : Yes                                       e
e                                                                        e
e ENTER to view details on the activation scope.                         e
e                                                                        e
DsssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssM
```

When you press Enter, the screen shown in Example 20-3 is displayed.

Example 20-3 Status of dynamic I/O reconfiguration readiness

```
Esssssssssssssssssssssssssssss Message List sssssssssssssssssssssssssssssN
e   Save  Query  Help                                                    e
e ------------------------------------------------------------------------  e
e                                                              Row 1 of 2 e
e Command ===> ____________________________________________ Scroll ===> CSR    e
e                                                                        e
e Messages are sorted by severity. Select one or more, then press Enter.      e
e                                                                        e
e / Sev Msg. ID  Message Text                                            e
e _ I   CBDA781I Your system configuration provides full dynamic              e
e #              reconfiguration capability.                             e
e ***************************** Bottom of data ****************************** e
```

By using the preceding process, you will then able to confirm whether your systems are in sync for a dynamic I/O reconfiguration change.

20.8 Duplicate device numbers

When merging an incoming system into the target sysplex, there is the possibility of a conflict of device numbers between the incoming system and the target sysplex. You will need to review the HCD I/O definitions for the incoming system and the target sysplex to ensure that any conflicting device numbers are resolved as part of the merge process.

Using HCD, you can generate various configuration reports which can then be used to compare the definitions of your incoming system and target sysplex to identify any duplicate device numbers conflict. When the reports have been generated, save the output to a data set and use a compare utility (such as ISPF Super-C) to identify any duplicate device numbers that may exist. Remember that not all duplicate device numbers are a problem—they are only a problem if the same device number is used to refer to two different actual devices.

20.9 NIP console requirements

Every system needs to have a destination to write the messages produced during NIP processing. The console used for this must be either attached to a non-SNA, channel-attached terminal controller (3x74 or a 2074, or, if none of these are available, NIP will use the system console function on the HMC. If the incoming system already has a NIP console, then no changes are required. If it does not, or if that console will be removed as part of the merge (perhaps because the sysplex limit of 99 consoles would have been exceeded), you can use the HMC as the NIPCONS. If you are using the HMC, that console is *not* defined in the NIPCONS field in HCD.

20.10 Eligible Device Table and esoterics

An Eligible Device Table (EDT) is a list of esoteric names. Each esoteric is a list of devices that can be accessed using that name. There can be more than one such list in each z/OS operating system definition, but only one may be in use at any one time in each operating system.

When merging an incoming system into the target sysplex, a review of esoterics needs to be carried out. The objective of the review is to ensure that work (batch jobs, started tasks, TSO users) will have access to the devices it needs after the merge, and bearing in mind where each unit of work can potentially run, without requiring any JCL changes. For example, if there is an esoteric called BIGDASD defined on the incoming system, and jobs from that system will potentially run on any system in the sysplex after the merge, you need to make sure that BIGDASD will be defined on every system, and will point at the correct devices.

You also need to check for clashing definitions. For example, if the esoteric called FASTTAPE points at tape devices on the incoming system, but at a pool of Tape Mount Management DASD on the target sysplex, you will need to decide how you will handle this after the merge.

You may also wish to take this opportunity to remove esoterics that are no longer in use. Should you decide to do this, however, you must make sure that none of those esoterics were used to catalog data sets. SYS1.SAMPLIB(IEFESOJL) contains a sample program to analyze the output from an IDCAMS LISTCAT command and report any data sets that have been cataloged via an esoteric.

You could also use the Catalog Search Interface (CSI) to retrieve this information from the catalogs. Catalog Search Interface (CSI) is a read-only general-use programming interface that is used to obtain information about entries contained in ICF Catalogs. Refer to SYS1.SAMPLIB(IGGCSI*) for sample programs on how to use the Catalog Search Interface.

20.11 Single or multiple OS Configs

The OS Configuration contains definitions for I/O devices, possibly NIP consoles, EDTs and esoterics. An IODF may contain multiple OS configurations.

The benefit of consolidating the OS configurations into a single definition is to simplify the management of the I/O configuration for your environment. If you need to change an OS configuration definition, you only have to change one OS configuration instead of carrying out the task multiple times. By doing the OS configuration change once, you also minimize any chances of errors being introduced if you avoid having to make the change multiple times against different OS configurations.

In a GoldPlex or a PlatinumPlex environment, you should definitely aim to have just a single OS configuration that is used by all systems in the sysplex. This delivers all the benefits just described, and also ensures that all systems will behave in a consistent manner in relation to the information contained in the OS configuration.

On the other hand, if your target environment will be a BronzePlex, the use of two OS configurations gives you the ability to have just one master IODF, but still be able to limit the devices that different members of the sysplex have access to.

20.12 ESCON logical paths

When merging a system(s) into a target sysplex environment, you will need connectivity to various peripheral hardware. You should use Multiple Image Facility (MIF) to share physical ESCON and/or FICON channels between LPARs on the same CPC, thereby reducing the number of channels required.

In addition to the requirement for physical connectivity to the I/O subsystem, however, you also need to be aware of the impact of any configuration changes on the ESCON logical path configuration. Different control units have different limits on the number of logical control units supported per logical control unit (LCU). If you have defined more paths than are supported by the control unit, you may encounter strange and confusing behavior whereby logical paths may not be able to be established. Logical paths are established at LPAR activation for any control unit that is on a channel that contains that LPAR in its access list, so the LPAR that is not able to establish a path will depend on the sequence in which the LPARs are activated.

To avoid this situation, you must determine how many paths are in use for each LCU today, and how many will be required after a configuration changes you plan to make. To obtain this information, you can use the ICKDSF ANALYZE command as shown in Example 20-4.

Example 20-4 Sample ICKDSF Job

```
//SOO1        EXEC PGM=ICKDSF
//DISK1       DD UNIT=SYSALLDA,DISP=SHR,VOL=SER=TOTMQF
//SYSPRINT    DD SYSOUT=*
//SYSIN       DD *
  ANALYZE DDNAME(DISK1) NODRIVE NOSCAN
/*
```

Example 20-5 Sample output from ICKDSF ANALYZE

```
LOGICAL   PATH   STATUS
 +----------------+---------+---------+---------+-----------------------------+
 ? LOGICAL PATH   ?         ?         ?         ? HOST PATH GROUP ID           ?
 ?                ? SYSTEM  ? HOST    ?         ?-------------+------+----------?
 ?---------+------? ADAPTER ? LINK    ? FENCED  ? CPU         ? CPU  ? CPU TIME ?
 ? NUMBER  ? TYPE ? ID      ? ADDRESS ?         ? SERIAL #    ? TYPE ? STAMP    ?
 +---------+------+---------+---------+---------+-------------+------+----------+
 ?1        ? E/L  ? 08      ? C90C    ?         ?             ?      ?          ?
 +---------+------+---------+---------+---------+-------------+------+----------+
 ?2        ? E    ? 08      ? C90C    ?         ?             ?      ?          ?
 +---------+------+---------+---------+---------+-------------+------+----------+
 ?3        ? E/L  ? 08      ? C90B    ?         ? 00000B0ECB ? 2064 ? B75CCB14 ?
 +---------+------+---------+---------+---------+-------------+------+----------+
 ?4        ? E    ? 08      ? C90B    ?         ? 00000B0ECB ? 2064 ? B75CCB14 ?
 +---------+------+---------+---------+---------+-------------+------+----------+
 ?5        ? E/L  ? 08      ? C90A    ?         ? 00000A0ECB ? 2064 ? B769816D ?
 +---------+------+---------+---------+---------+-------------+------+----------+
 ?6        ? E    ? 08      ? C90A    ?         ? 00000A0ECB ? 2064 ? B769816D ?
 +---------+------+---------+---------+---------+-------------+------+----------+
 ?7        ? E/L  ? 08      ? C909    ?         ? 0000090ECB ? 2064 ? B7697F28 ?
 +---------+------+---------+---------+---------+-------------+------+----------+
 ?8        ? E    ? 08      ? C909    ?         ? 0000090ECB ? 2064 ? B7697F28 ?
 +---------+------+---------+---------+---------+-------------+------+----------+
 ?9        ? E/L  ? 08      ? C908    ?         ? 0000080ECB ? 2064 ? B7697AFF ?
 +---------+------+---------+---------+---------+-------------+------+----------+
```

The output from the ANALYZE command (shown in Example 20-5) tells you which paths are established, which host adapter each is associated with, and the CPU serial number and partition number of the host that is using each logical path.

Restriction: The ICKDSF ANALYZE command does not work for RAMAC® Virtual Arrays (9393).

If your environment consists of IBM and non-IBM hardware, you will need to be in liaison with the hardware support representative to verify that you will not exceed the logical paths for the respective control unit.

20.12.1 DASD control unit considerations

Regardless of the DASD configuration in your target sysplex or incoming system, the issue of logical paths still remains. You must ensure that the data that you need to access from systems within the target sysplex are stored on DASD devices that are connected through control units that provide sufficient logical paths.

The Enterprise Storage Server® (2105) supports a total of 2048 logical paths, 128 per logical control unit, with a maximum of 64 per ESCON port.

The IBM 3990-6 supports up to 128 logical paths, 16 per ESCON port. Similar limits apply to the 9390 control unit.

The IBM RAMAC Virtual Array Model 2 subsystem supports up to 128 logical paths.

20.12.2 Tape control unit considerations

The 3590-A50 tape subsystem can have up to two ESCON adapters, each with 64 logical paths. The 3590-A60 tape subsystem can have up to eight ESCON adapters, each with 64 logical paths, providing a total of up to 512 logical paths.

Existing IBM 3490/3490E tape drives may be upgraded with ESCON channel interfaces (if not already done) to provide 16 logical paths per ESCON port.

The ESCON-attached 3494 Virtual Tape Server (VTS) looks like a 3490E control unit to ESCON, therefore, the number of logical paths are the same as the standard 3490E.

20.12.3 Non-IBM hardware

If your target sysplex and incoming system configuration contains hardware from other suppliers, you need to ensure that they provide sufficient physical and logical paths to enable the connectivity you require.

It is also normal practice to ensure that all control units (IBM and OEM) are at an appropriate microcode level before commencing a merge. You will also need to verify that the systems to which devices are to be accessed have the relevant Unit Information Modules (UIMs) installed to correctly support the devices.

20.12.4 Switch (ESCON/FICON) port considerations

If you have ESCON and/or FICON switch(es) as part of your configuration, you will need to determine if there are sufficient ports installed to cater for the additional connectivity requirements of the incoming system. You will need to liase with your hardware representative to assess whether there are sufficient physical ports available on your switch(es) to accommodate the additional connectivity requirements.

If your target sysplex and incoming system have I/O Operations or ESCON Manager installed, and the port names in the directors are kept up to date, you can use that product to query the switch to look for unassigned and offline ports.

We recommend that as part of the merge task, you liase with your hardware representative and carry out a physical audit on your incoming system and target sysplex environment to ensure that there HCD and your configuration diagrams accurately reflect the actual configuration.

20.13 CF connectivity

Connectivity of all systems in the target sysplex to the CF (integrated or external) will need to be considered.

The connectivity from an operating system to a CF and vice versa is provided by coupling links. z/OS and CF images may be running on the same or on separate machines. Every z/OS image in the target sysplex must have *at least* one coupling link to each CF image.

For availability reasons, there should be:

- At least two coupling links between z/OS and CF images
- At least two CFs (not running on the same CPC)
- At least one standalone CF (unless you plan on using System-Managed CF Structure Duplexing)

There are several types of coupling links available, and their support will depend on the type of CPC(s) you have installed in your target sysplex environment.

The types of coupling links available are:

- 9672
  - IC (connection within the same CPC only)
  - ICB (connection up to 10 metres)
  - ISC (connection up to 40 km)
- 2064 (z900)
  - IC3 (peer IC, connection within the same CPC only)
  - ICB (connection up to 10 metres - compatible with ICB on 9672)
  - ICB3 (peer ICB, connection up to 10 metres - zSeries only)
  - ISC (connection up to 40 km - compatible with ISC on 9672)
  - ISC3 (peer ISC, connection up to 40km - zSeries only)
- 2066 (z800)
  - IC3 (peer IC, connection within the same CPC only)
  - ICB3 (peer ICB, connection up to 10 metres - zSeries only)
  - ISC3 (peer ISC, connection up to 40km - zSeries only)
  - ISC (connection up to 40 km - compatible with ISC on 9672)

If your target sysplex environment has a mixed processor type configuration, you will need to ensure that you have the correct coupling links installed.

20.14 FICON/ESCON CTCs

Channel-to-channel (CTC) control units and associated devices provide the infrastructure for intersystem communications. z/OS exploiters of CTC communications include:

- XCF using Pathin and Pathout devices for intersystem communication between z/OS images
- VTAM
- TCP/IP

20.14.1 ESCON CTC

CTC support in the ESCON environment is provided with a pair of connected ESCON channels. These channels could be configured either in a point-to-point or switched point-to-point configuration, where one ESCON channel is defined as TYPE=CNC and the other ESCON channel is defined as TYPE=CTC. Both shared (MIF) and non-shared ESCON channels support the ESCON CTC function.

Attention:

- The ESCON channel defined as CTC can only support the ESCON CTC Control Unit (CU) function and CTC devices. It cannot be used for other I/O device support, such as disk, tape etc.
- The ESCON channel defined as CNC can support ESCON CTC control unit definitions and other I/O control unit type definitions, such as disk and tape, if the channel is connected in a *switched* point-to-point topology.

20.14.2 FICON CTC

Channel-to-channel communication in a FICON environment is provided between two FICON (FC) channel FCTC control units, with at least one of the two FCTC Control Units being defined on an FC channel on a zSeries processor at driver level 3C or later. The other FCTC CU is defined on a FICON (FC) channel on any of the following processors:

- 9672 G5/G6 CPC
- z800 CPC
- z900 CPC

20.14.3 Differences between ESCON and FICON CTC

There are several differences between the ESCON and FICON CTC implementations, as shown in Table 20-2.

Table 20-2 Characteristics of ESCON and FICON CTCs

Characteristic	ESCON	FICON
Number of required channels	At least 2	1 or 2
Channel dedicated to CTC function	Yes (1 of the 2)	No
Number of unit addresses supported	Up to 512	Up to 16384
Data transfer bandwidth	12-17 MB/sec	60-90+ MB/sec
Number of concurrent I/O operations	1	Up to 32
Data transfer mode	Half duplex	Full duplex

The details of these differences are as follows:

- ESCON CTC connectivity is provided by a pair of ESCON channels - one is defined as CTC and the other defined as CNC. At least two ESCON channels are required.

  FICON CTC connectivity can be implemented using one or two FICON (FC) native channels.

- An ESCON channel defined as CTC can only support the CTC function. Only an SCTC control unit can be defined on an ESCON CTC channel.

  The FICON native (FC) channel supporting the FCTC control unit can communicate with an FCTC control unit on either the z800, z900, or 9672 G5/G6 CPC and at the same time, the same FICON (FC) channel can also support operations to other I/O control unit types such as disk and tape.

- An ESCON CTC channel supports a maximum of 512 device addresses.

  A FICON native (FC) channel supports a maximum of 16,384 device addresses.

- An ESCON channel has a data transfer bandwidth of 12-17 MB, significantly less than the 60-90+ MB for a FICON channel.

- An ESCON channel supports only one actively communicating I/O operation at a time. The FICON channel supports up to 32 concurrent I/O operations.

- An ESCON channel operates in half-duplex mode, transferring data in one direction only at any given time. A FICON channel operates in full duplex mode, sending and receiving data at the same time.

20.15 Sysplex Timer

The IBM Sysplex Timer synchronizes the processor's time-of-day clock in multiple systems. The Sysplex Timer provides an external time source for all attached processors. You need a Sysplex Timer installed to support the configuration when multiple CPCs will be coupled in a sysplex.

Consideration will also need to be given to how to handle the situation if the systems within the target environment will be using different time zones.

20.15.1 LPAR Sysplex ETR support

To allow you to run different sysplexes on the same hardware, with each sysplex using a different time zone, you can use the LPAR Sysplex ETR Support. Also, via a common ETR, the LPAR Sysplex ETR Support function allows you to attach separate CPCs, operating at different local TODs, to operate off the same ETR.

20.15.2 Parmlib

If you will be using a Sysplex Timer, you must ensure that the CLOCKxx member of Parmlib is configured correctly and that all systems will be using the same CLOCKxx member (or at least, that all members are using a CLOCKxx with the same statements).The CLOCKxx member for a system that is a member of a multisystem sysplex must contain a specification of ETRMODE YES. The system then uses the Sysplex Timer to synchronize itself with the other members of the sysplex. The system uses a synchronized time stamp to provide appropriate sequencing and serialization of events within the sysplex.

More information about the interaction between the Sysplex Timer and the systems in a sysplex is available in the IBM Redbook entitled *S/390 Time Management and IBM 9037 Sysplex Timer*, SG24-2070.

20.16 Console connectivity

As discussed in 19.2, "Consoles" on page 311, the requirement to have console connectivity to all systems in the target sysplex is not essential.

You may consider undertaking a console consolidation exercise prior to merging the incoming system and rationalize your existing console environment based on operational function. For example, you may have a console in your tape library area that could be configured so that it only handled Route Code 3 (tape library pool) messages for all of the systems in the target sysplex.

As of z/OS 1.1, SNA MCS consoles were introduced. SMCS consoles are MCS consoles that use VTAM services for input and output. SMCS is a VTAM application and is therefore reliant on VTAM being available for communication.

You may also want to consider the use of Extended MCS (EMCS) consoles. Several products (TSO/E, SDSF, System Automation/390) offer this capability.

20.17 Hardware Management Console (HMC) connectivity

The Hardware Management Console (HMC) is a central point for the management of the CPCs in your target sysplex configuration. You will need to ensure that the driver microcode level on the HMC(s) will support the incoming system CPC(s), as a down-level HMC cannot support an up-level processor. You will need to also configure an HMC so that it is enabled for Licensed Internal Code (LIC) changes.

You will also need to decide whether your HMC will be used for additional applications, such as sysplex timer and/or the ESCON director. Refer to the relevant product manuals for further information.

The HMC is discussed in 19.1, “One or more HMCplexes” on page 302.

20.18 Tools and Documentation

In this section, we provide information about tools and documentation that may help you complete this task.

20.18.1 Tools

The following tools may be of assistance for this merge task:

Parallel Sysplex Coupling Facility Structure Sizer (CFSizer)

The Parallel Sysplex Coupling Facility Structure Sizer is a Web-based wizard that can assist you in verifying the correct sizing of your CF structures.

http://www.ibm.com/servers/eserver/zseries/cfsizer/

XISOLATE

The XISOLATE tool can be used to assist you in identifying if there are single points of failure affecting critical data sets that you specify. More information about XISOLATE can be found at:

http://www.ibm.com/servers/eserver/zseries/pso/

IEFESOJL

This is a sample program provided in SYS1.SAMPLIB that is used to analyze the output from an IDCAMS LISTCAT command, and report on any data sets that have been catalogued using an esoteric.

Catalog Search Interface (CSI)

Catalog Search Interface (CSI) is a read-only general-use programming interface that is used to obtain information about entries contained in ICF Catalogs. There are several sample programs (IGGCSI*) provided in SYS1.SAMPLIB.

ICKDSF

Using the ANALYZE command of ICKDSF, you can obtain a logical path status report for your DASD subsystems.

20.18.2 Documentation

The following publications provide information that may be of assistance for this merge task:

- *FICON CTC Implementation*, REDP0158
- *Hardware Configuration Definition (HCD) Planning,* GA22-7525
- *Hardware Configuration Definition (HCD) Scenarios,* SC33-7987
- *MVS/ESA HCD and Dynamic I/O Reconfiguration Primer,* SG24-4037
- *OS/390 Parallel Sysplex Configuration Volume 3: Connectivity,* SG24-5639
- *z/OS Hardware Configuration Definition (HCD) Users Guide*, SC33-7988
- *zSeries 900 ESCON Channel-to-Channel Reference*, SB10-7034
- *zSeries 900 System Overview,* SA22-1027
- *zSeries 900 Technical Guide,* SG24-5975
- Coupling Facility Level information, available on the Web at:
  `http://www.ibm.com/servers/eserver/zseries/pso/`

21

System exits and usermod considerations

This chapter discusses the considerations for system exits and usermods when moving an existing system into an existing sysplex.

21.1 Overview

There are several reasons to rationalize system modifications as part of the sysplex merge process. If the libraries containing the usermods and exits are shared—for example, on a shared sysres, then the rationalizing of those modifications is a requirement for the sysplex merge.

Even if the libraries are not shared, the modification may affect processing on other systems in the sysplex. In a JES2 MAS environment, for example, converter/interpreter processing can occur on any system within the MAS. A job submitted on one system within the MAS, may be interpreted on another system and execute on a third system. Any exits executed during that process must act in a consistent and coherent manner across the sysplex.

Aside from the technical issues, there are the maintenance considerations. Maintaining multiple copies of exits makes it difficult to build systems from a common base. Multiple software bases incur larger support overheads. Part of sysplex merge planning should include a full review of all system modifications. Only then can an informed decision be made as to when and how the modifications need to be handled.

21.2 Definitions

An *exit* is a defined interface point, where the operating system or program product will call a user-written program to make some sort of decision. Exits are usually written in assembler and developed at a user installation.

A *usermod* is a modification to the operating system or support software. A usermod can range from a simple modification of an ISPF panel, to changes to JES2 source code, to zaps to system modules.

21.3 Simplicity

Any modifications to the operating system and support software should be avoided wherever possible. The bulk of the time spent installing a new release is retrofitting exits and products that are not supplied on the ServerPac—far more than the time that is spent testing.

Also, a significant amount of time is required to test and debug operating system modifications. When debugging system software problems, exits and usermods are always suspect. A handy rule of thumb is “if the problem you are experiencing has not already been reported in IBMLINK, then it is a good idea to check if local exits or usermods could be involved”.

Operating system exit points are maintained from release to release. There is a clearly defined interface. The exit may need recompiling when migrating to another software release, but it is unlikely that the interface will change.

Usermods, however, are not as robust. There is no defined interface, so even minor product code changes can force a change to a usermod. Usermods are usually applied manually, which can be a time-consuming task, especially if the modification is extensive. Even if SMP/E is used to make the modification, refitting the modification can be tedious. Panel updates, for example, cannot be packaged as updates. A full replacement usermod is required. A refit of this type of user mod requires extraction of the modification from the “old” panel and integrating it into the updated version of the panel.

21.4 Reviewing system modifications

The first thing you must do is to assemble an inventory of all your exits and usermods. Not only can you not do a comprehensive review without such a list, you will also spend more time every time you upgrade your system as you struggle to find and address al the exits and mods.

Once you have a complete list, the next thing to do is to examine each exit or system modification critically. Is the function still required? It is not uncommon to find that the business reason for a particular exit or usermod has disappeared, but no one told the systems programmers! As a result, time is spent (wasted) on every upgrade refitting an exit that is no longer needed.

If you determine that the function provided by the exit or usermod *is* required, the next step is to investigate if the product now contains the same or similar function to your exit. Removing unnecessary exits will not only make the sysplex merge procedure simpler, it will also reduce the ongoing maintenance requirement for the sysplex.

Finally, if you determine that a particular exit or usermod is still required, compare the functions of the target sysplex version of the exit and the exit running on the incoming system. Can a master version of the exit be developed, which will contain all functions of both exits?

To generalize the exit, and make ongoing maintenance easier, you should consider externalizing any variables or hard-coded values.For example, these can be stored in RACF profiles, with a system name as part of the profile name. This will allow an exit to return different values on each system in the sysplex while still using the same exit code and sharing the same RACF database.

It is wise to minimize differences between how the exits behave on different systems in the sysplex if possible. If absolutely necessary, different paths through the exit could be executed depending on which the system the exit is running on. From within an exit, the CVTSNAME field of the CVT can be used to identify the system name.

Warning: Having exits that perform differently on different systems could impact your ability to exploit the resource sharing capability of sysplex. For example, an IEFUSI SMF exit, which assigned different region sizes on different systems could impact your ability to move work from one system to another.

Some commonly modified product options to be considered are contained in Table 21-1.

Table 21-1 Products containing execution options

Product	Notes
DFSORT options	
PL/I compiler and run-time options	See note[a]
COBOL compiler and run-time options	See note[a]
C compiler and run-time options	See note[a]
C++ compiler and run-time options	See note[a]

a. See Chapter 9, “Language Environment considerations” on page 135

21.5 Supporting system modifications

If the exit or usermod is essential, there are two requirements that need to be met to support that modification: good documentation and the ability to track the modification.

21.5.1 Documentation

It is important to document the reason for the modification, who requested it and when, and what issue the modification addresses. Often the reason for a particular modification is lost over time, and the modification is endlessly refitted as new software releases are installed. Good documentation will allow a review on each upgrade of the applicability of all modifications.

Often new operating system features can obviate the need for a modification. Software is enhanced, partly, based on user experiences and feedback. So, it is common for later releases to contain the same or similar functionality as many user modifications. If you have found a shortcoming with a particular function, the chances are that other users will have too.

21.5.2 Tracking

It is essential to install and maintain system exits or modifications in a manner that is trackable. It is all too easy to simply edit an ISPF panel definition to make a change, only to have it “accidentally” regressed some time later, when maintenance is applied. And regression is not the only possible consequence.

Consider the following example: Imagine you modify an OPC panel. If you make the change in the OPC-provided library, a subsequent PTF could overwrite your modification, regressing your change without your knowledge. On the other hand, if you copy the modified panel into a library higher up the ISPPLIB concatenation, then when you apply the PTF, you are not actually picking up all of the PTF because you are (probably unknowingly) running off an old copy of the panel. Neither situation is pleasant, and not at all uncommon.

To avoid this situation, all system modifications should be made with SMP/E. This provides a site-independent method of maintaining system modifications. Support personnel unfamiliar with the documentation conventions of a particular site can still maintain a modification if SMP/E is used.

All *source code* for system exits should be packaged as an SMP/E Usermod, even if the particular exit does not have SMP/E definitions for the source code. With the source code packaged with the usermod, there can be no doubt that the source does indeed match the installed load module. The copy of the usermod in the SMPPTS can be viewed if there is any doubt as to the contents of an exit.

Example 21-1 shows a sample SMP/E Usermod that can be used as a skeleton to install an exit using source code. During SMP/E apply processing, the source element named in the ++SRC statement will be added or replaced. SMP/E will then assemble and linkedit the module into the target.

Example 21-1 Skeleton SMP/E usermod

```
//         EXEC  PGM=GIMSMP,REGION=6M
//SMPCSI   DD    DSN=SMPE.OS390.GLOBAL.CSI,DISP=SHR
//SMPCNTL  DD    *
  SET    BOUNDARY (GLOBAL).
  RECEIVE S (umod001) SYSMODS.
  APPLY S (umod001).
/*
```

```
//SMPPTFIN DD DATA
++USERMOD(umod001) REWORK(yyyydddn).
++VER(Z038) FMID(fmid) PRE(prereqs).
++SRC(member).
    Exit source code
/*
```

Note: A ++JCLIN statement and linkedit JCL will be required if the exit is not already defined to SMP/E. This can be checked using the SMP/E dialog. If the target zone does *not* contain MOD and LMOD entries for the exit, then a JCLIN is required.

21.6 Documentation

The following documents contain information that may assist you in identifying and managing user exits and user modifications on your system:

- *OS/390 MVS Installation Exits,* SC28-1753
- *Parallel Sysplex - Managing Software for Availability,* SG24-5451
- *SMP/E z/OS and OS/390 Reference,* SA22-7772
- *SMP/E z/OS and OS/390 User's Guide,* SA22-7773
- *z/OS MVS Installation Exits,* SA22-7593
- *z/OS MVS Setting Up a Sysplex,* SA22-7625
- *z/OS ISPF Planning and Customizing,* GC34-4814
- *z/OS JES2 Installation Exits,* SA22-7534

In addition, there is a non-IBM product called "Operating System/Environment Manager" that eases the task of generating and maintaining system exits. More information about this product can be found at:

http://www.triserv.com/os-em-in.html

21.7 Useful exits and usermods

The following are some useful exits and usermods to consider when merging a system into a sysplex. The modifications presented here are intended to be used to aid the sysplex merge process. They are only useful in certain circumstances, depending on the configuration of the resulting sysplex. A PlatinumPlex does not require any of these modifications.

When merging dissimilar systems into a sysplex, in order to minimize the amount of change introduced, intermediate steps may be required. It is often easier to maintain elements of the existing system separation than attempt to merge everything in a single step. This is where these sample modifications may be useful.

21.7.1 Duplicate TSO logons in a JES2 MAS

Prior to z/OS 1.4, JES2 does not allow duplicate TSO logons within the same JES2 MAS, even though the TSO user is logging on to different systems. It is often a requirement for support personnel to be able to log on to multiple systems within a sysplex concurrently, particularly when systems within the sysplex are configured differently.

To allow duplicate logons, a small source code change is required to the HASPCNVT module of JES2 as shown in Example 21-2.

Example 21-2 Sample JES2 Usermod

```
//MJ2OZ22 JOB TPSCP010,MJ2OZ22,CLASS=E,MSGCLASS=H,
//         NOTIFY=&SYSUID
//*--------------------------------------------------------------------*
//*                                                                    *
//* USERMOD:  MJ2OZ22                                                  *
//*                                                                    *
//* FUNCTION: ALLOW DUPLICATE TSO LOGON'S IN A MAS                     *
//*                                                                    *
//*--------------------------------------------------------------------*
//*
//SMPE     EXEC  PGM=GIMSMP,REGION=6M
//SMPCSI   DD DSN=SMPEG15.OS390.GLOBAL.CSI,DISP=SHR
//SMPCNTL  DD *
  SET BOUNDARY(GLOBAL) .
    RECEIVE S(MJ2OZ22) SYSMODS .
  SET BOUNDARY(J2O2AOT) .
    APPLY S(MJ2OZ22) REDO .
/*
//SMPPTFIN DD DATA
++USERMOD(MJ2OZ22) REWORK(20012781).
++VER(Z038) FMID(HJE7703) PRE(UW74954) .
++SRCUPD(HASPCNVT) .
./ CHANGE NAME=HASPCNVT
*        JZ    XTDUPEND           Skip duplicate logon check  @42OP190 05991200
         J     XTDUPEND           Allow duplicate logon in MAS MJ2OZ22 05991201
/*
```

TSO uses the SYSIKJUA enqueue major name to prevent duplicate logons. If you want to be able to log on more than once in a MAS, this enqueue must not be propagated around the sysplex. The default scope for this enqueue is SYSTEM, so ensure that the following entry is *not* part of the GRSRNLxx member in use:

```
RNLDEF RNL(INCL) TYPE(GENERIC) QNAME(SYSIKJUA)
```

21.7.2 ISPF Exit 16: Log, List, and Temporary Data Set Allocation Exit

If a user needs to be able to log onto multiple systems in sysplex concurrently using the same logon id, then consider using ISPF exit 16.

By default, ISPF generates names for list and log data sets which are not unique within a sysplex. A TSO user attempting to use ISPF on more than one system in a sysplex concurrently will encounter data set enqueue conflicts.

ISPF Exit 16 can be used to provide a prefix to be added to the default ISPF-generated data set names. The exit should ensure the prefix yields unique data sets names on each system in the sysplex. Adding the system name, for example, as a second or third level qualifier will create unique names.

For example, ISPF data sets for system PLX01 would be called:

```
USERID.PLX01.HFS
USERID.PLX01.ISPPROF
USERID.PLX01.ISR0001.BACKUP
USERID.PLX01.RMF.ISPTABLE
USERID.PLX01.SPFLOG1.LIST
```

```
   USERID.PLX01.SPF120.OUTLIST
   USERID.PLX01.SPF3.LIST
```

The sample exit shown in Example 21-3 will add the system name to the end of the ISPF-generated data set name prefix for all ISPF list and log data sets. The sample assumes a system name length of 5 bytes. The code should be modified for other, or variable length, system names.

For details concerning the installation of this exit, consult *Interactive System Productivity Facility (ISPF) Planning and Customization,* SC28-1298.

Example 21-3 ISPF Exit 16

```
//MOS#Z21 JOB ACCT,MOS#Z21,CLASS=E,MSGCLASS=H,
//            NOTIFY=&SYSUID
//*--------------------------------------------------------------------*
//*                                                                    *
//* USERMOD:  MOS#Z21                                                  *
//*                                                                    *
//* FUNCTION: ADD SYSTEM NAME TO ISPF LIST/LOG/SPFTEMP DATASETS        *
//*                                                                    *
//*--------------------------------------------------------------------*
//*
//MOS#Z21  EXEC  PGM=GIMSMP,REGION=6M
//SMPCSI   DD    DSN=SMPEG15.OS390.GLOBAL.CSI,DISP=SHR
//SMPCNTL  DD    *
  SET    BOUNDARY (GLOBAL).
    RECEIVE S (MOS#Z21) SYSMODS .
  SET    BOUNDARY (OS#2A0T) .
    APPLY S (MOS#Z21)       .
/*
//SMPPTFIN DD DATA
++USERMOD (MOS#Z21) REWORK(20010500) .
++VER (Z038)  FMID(HIF5A02).
++JCLIN .
//LKED1  EXEC PGM=IEWL,PARM='LIST,XREF,REUS,RENT'
//SYSPRINT DD SYSOUT=A
//SYSLMOD  DD DSN=SYS1S2A.SISPLOAD,DISP=SHR
//SYSUT1   DD UNIT=SYSDA,SPACE=(CYL,(3,1))
//SYSLIN   DD *
  ORDER  ISPXDT
  ENTRY  ISPXDT
 INCLUDE AUSERMOD(ISPFUX16)
 INCLUDE AISPMOD1(ISPXDT)
 NAME ISPEXITS(R)
++SRC (ISPFUX16) DISTLIB(AUSERSRC).
ISPFUX16 CSECT
ISPFUX16 AMODE 31
ISPFUX16 RMODE 24
         BAKR  R14,0              STACK SYSTEM STATE
         LR    R12,R15            GET EXIT ENTRY ADDR
         USING ISPFUX16,R12       ESTABLISH BASE ADDR
         L     R2,28(0,R1)        GET PREFIX ADDR
         L     R3,24(0,R1)        GET LENGTH ADDR
         L     R5,0(0,R3)         LENGTH OF PREFIX
         LA    R4,0(R5,R2)        POINT TO END OF PREFIX
         MVI   0(R4),C'.'         ADD PERIOD FOR NEW QUALIFIER
         L     R7,CVTPTR          GET CVT ADDR
         USING CVT,R7             MAP CVT
         MVC   1(5,R4),CVTSNAME   ADD SYSTEM NAME
```

```
         DROP  R7                 DROP CVT
         LA    R5,6(0,R5)         CALC NEW PREFIX LENGTH
         ST    R5,0(0,R3)         SAVE NEW PREFIX LENGTH
         SR    R15,R15            SET ZERO RETURN CODE
         PR                       UNSTACK SYSTEM STATE
*
         REGEQU
         CVT   LIST=NO,DSECT=YES
         END
++SRC (ISPXDT) DISTLIB(AUSERSRC).
         ISPMXED START
         ISPMXLST (16)
         ISPMXDEF 16
         ISPMEPT ISPFUX16
         ISPMXEND
         ISPMXED END
         ISPMXDD START
         ISPMXDD END
         END
/*
```

Note: A user's ISPF profile data set should be allocated with a DISP=OLD. Thus, each instance of a TSO user's session should use a different ISPPROF data set. This can be done as part of the logon process, usually in the initial clist executed from the logon procedure.

21.7.3 PDF Data Set Name Change Exit

A similar problem exists for Edit recovery data sets. If a user is logged on more than once in a sysplex, the allocation of edit recovery data sets will attempt to allocate non-unique data set names within a sysplex. This will fail, and edit recovery will not be available. To prevent this, the PDF Data Set Name Change Exit can be used.

Prior to the allocation of edit recovery data sets, the PDF Data Set Name Exit is called. This exit may change the name of the data set about to be allocated. The example exit in Example 21-4 will insert the system name after the second level qualifier of the data set name.

For details concerning the installation of this exit, consult *Interactive System Productivity Facility (ISPF) Planning and Customization,* SC28-1298.

Example 21-4 PDF Data Set Name Change Exit

```
//MOS#Z20 JOB TPSCP010,MOS#Z20,CLASS=E,MSGCLASS=H,
//          NOTIFY=&SYSUID
//*--------------------------------------------------------------------*
//*                                                                    *
//* USERMOD:  MOS#Z20                                                  *
//*                                                                    *
//* FUNCTION: ADD SYSTEM ID TO ISPF WORK DATASET NAMES                 *
//*                                                                    *
//*--------------------------------------------------------------------*
// *
//MOS#Z20  EXEC  PGM=GIMSMP,REGION=6M
//SMPCSI   DD    DSN=SMPEG15.OS390.GLOBAL.CSI,DISP=SHR
//SMPCNTL  DD    *
  SET    BOUNDARY (GLOBAL).
  RECEIVE S (MOS#Z20) SYSMODS.
```

```
  APPLY S (MOS#Z20) REDO.
/*
//SMPPTFIN DD DATA
++USERMOD (MOS#Z20) REWORK(20010500).
++VER (Z038)  FMID(HIF5A02) .
++JCLIN .
//LKED1  EXEC PGM=IEWL,PARM='LIST,XREF,REUS,RENT,REFR'
//SYSPRINT DD SYSOUT=A
//SYSLMOD  DD DSN=SYS1S2A.SISPLPA,DISP=SHR      <== DDDEF
//SYSUT1   DD UNIT=SYSDA,SPACE=(CYL,(3,1))
//SYSLIN   DD *
INCLUDE AUSERMOD(ISRNAMEC)
NAME ISRNAMEC(R)
INCLUDE AUSERMOD(ISRNOGEN)
NAME ISRNOGEN(R)
++SRC (ISRNAMEC) DISTLIB(AUSERSRC).
*--------------------------------------------------------------------*
*    THIS EXIT WILL INSERT THE SYSTEM NAME AFTER THE 2ND             *
*    QUALIFIER  IN THE DSNAME OF PDF WORK DATASETS                   *
*--------------------------------------------------------------------*
ISRNAMEC CSECT
ISRNAMEC AMODE 31
ISRNAMEC RMODE 24
         BAKR  R14,0               STACK SYSTEM STATE
         LR    R12,R15             GET ENTRY ADDR
         USING ISRNAMEC,R12        ESTABLISH BASE
         L     R2,20(0,R1)         GET REASON ADDR
         CLC   =C'TEMP',0(R2)      IS IT A WORK DATASET ?
         BE    INSERT              YES. BRANCH
         CLC   =C'RECOVERY',0(R2)  IS IT A WORK DATASET ?
         BNE   EXIT                NO. THEN EXIT
INSERT   DS    0H
         GETMAIN RU,LV=44          GET SPACE TO SAVE DSNAME
         LR    R11,R1              SAVE ADDR
         LR    R9,R1               SAVE ADDR
         EREG  R1,R1               RESTORE PARM REG
         L     R3,16(0,R1)         GET ADDR OF DSNAME
         MVC   0(44,R11),0(R3)     SAVE DSNAME
         LA    R10,44              DSNAME 44 CHARS LONG
         LA    R4,2                INSERT AFTER 2ND QUALIFIER
SCAN     DS    0H
         LA    R3,1(0,R3)          NEXT CHAR OF DSNAME
         LA    R11,1(0,R11)        NEXT CHAR OF DSNAME
         BCTR  R10,0               COUNT NUMBER OF CHARS LEFT
         CLI   0(R3),C'.'          NEW QUALIFIER ?
         BNE   SCAN                NO. LOOP UNTIL NEXT QUALIFIER
QUAL     DS    0H
         BCT   R4,SCAN             BRANCH UNTIL 2ND QUALIFIER
         L     R8,CVTPTR           GET CVT ADDRESS
         USING CVT,R8              MAP CVT
         MVC   1(3,R3),CVTSNAME    INSERT SYSTEM NAME IN DSNAME
         DROP  R8                  DROP CVT
         LA    R3,4(0,R3)          POINT PAST SYSTEM ID
         S     R10,=F'4'           ALLOW FOR SYSID LENGTH
         EX    R10,MOVEREST        MOVE IN REST OF DSNAME
         FREEMAIN RU,LV=44,A=(R9)  FREE STORAGE
EXIT     DS    0H
         SR    R15,R15             CLEAR R15
         PR                        UNSTACK SYSTEM STATE
*
```

```
MOVEREST MVC   0(0,R3),0(R11)       MOVE IN REST OF DSNAME
         LTORG
         REGEQU
         CVT   LIST=NO,DSECT=YES
         END
/*
```

Note: The PDF Data Set Name Change Exit is called for more than just edit recovery data set allocation. The sample exit processes only those data sets used for edit recovery by checking the allocation reason parameter passed to the exit. If the reason is RECOVERY, then the data set is the edit recovery data set. If the reason is TEMP, then the data set is a temporary PDF data set. All other data set names are ignored

For full details refer to *Interactive System Productivity Facility (ISPF) Planning and Customization,* SC28-1298.

21.7.4 JES2 Exit 4

The JES2 default system affinity for batch jobs is “ANY”. If the incoming system is to become a member of the target sysplex’s MAS, then by default, batch work is eligible for execution on any member of the MAS. Batch jobs can be confined to specific systems using job classes or scheduling parameters, but this often requires a large number of JCL changes.

System affinity can be forced for all jobs submitted on a particular system via the **$TINTRDR,SYSAFF=*** JES2 command. However, jobs submitted via Network Job Entry (NJE) or Remote Job Entry (RJE) will not have any system affinity assigned, unless a JOBPARM statement is included in the job specifying affinity to a system.

The JES2 Exit 4 sample exit shown in Example 21-5 will insert a JES2 JOBPARM system affinity statement in any job containing a JES2 route execute statement. Any jobs submitted remotely will have a system affinity forced.

Example 21-5 JES2 Exit 4

```
//MJ2OZO4 JOB TPSCPO1O,MJ2OZO4,CLASS=E,MSGCLASS=H,
//           NOTIFY=&SYSUID
//*-------------------------------------------------------------------*
//*                                                                   *
//* USERMOD:  MJ2OZO2                                                 *
//*                                                                   *
//* FUNCTION: ADD SYSTEM AFFINITY STATEMENTS                          *
//*                                                                   *
//*-------------------------------------------------------------------*
//*
//SMPE     EXEC  PGM=GIMSMP,REGION=6M,TIME=12O
//SMPCSI   DD DSN=SMPEG15.OS39O.GLOBAL.CSI,DISP=SHR
//SMPCNTL  DD *
  SET BOUNDARY(GLOBAL) .
    RECEIVE S(MJ2OZO4) SYSMODS .
  SET BOUNDARY(J2O2AOT).
    APPLY S(MJ2OZO4) REDO.
/*
//SMPPTFIN DD DATA,DLM=$$
++USERMOD(MJ2OZO4) REWORK(2OO1O5O1).
++VER(ZO38) FMID(HJE77O3).
++JCLIN .
//JOBNAME  JOB ,MSGLEVEL=(1,1)
//LINK     EXEC PGM=IEWL,PARM=(RENT,REFR,REUS,AMODE=31,RMODE=24)
//SYSLMOD  DD   DSN=SYS1.SHASLINK,DISP=SHR
```

```
//SYSLIN   DD *
 INCLUDE SYSPUNCH(JES2X04)
 ENTRY   JES2X04
 NAME    JES2X04(R)
/*
++SRC(JES2X04) DISTLIB(USERSRC) .
JES2X04 TITLE 'JES2 USER EXIT 4-- PROLOG (MODULE COMMENT BLOCK)'
*********************************************************************
*                                                                   *
* MODULE NAME : JES2X04                                             *
*                                                                   *
* DESCRIPTIVE NAME : JES2 USER EXIT 4  -  JCL/JECL SCAN ROUTINE     *
*                                                                   *
* FUNCTION: THIS EXIT CAPTURES THE SYSTEM NAME FROM THE /*XEQ CARD  *
*           AND INSERTS A SYSTEM AFFINITY JOBPARM STATEMENT         *
*                                                                   *
* DESCRIPTION:                                                      *
*                                                                   *
*                                                                   *
* ENVIRONMENT : JES2 MAIN TASK                                      *
*                                                                   *
* NOTES                                                             *
*                                                                   *
*    DEPENDENCIES = JES2 $EXIT FACILITY, STANDARD JES2 SERVICES     *
*                                                                   *
*    REQUIREMENTS = THIS CODE REQUIRES THE USE OF JCTUSER3 AND 4.   *
*                   IF THIS IS NEEDED FOR ANY OTHER PURPOSE, CHANGE *
*                   THIS EXIT TO USE SOME OTHER AVAILABLE JCT       *
*                   USER FIELD.                                     *
*                                                                   *
*    ATTRIBUTES = NON-REENTRANT, RMODE(ANY),AMODE(24/31)            *
*                                                                   *
* ENTRY POINTS : EXIT4                                              *
*                                                                   *
* REGISTER USAGE (ENTRY/EXIT) :                                     *
*                                                                   *
*    REG       VALUE ON ENTRY            VALUE ON EXIT              *
*                                                                   *
*    R0        A CODE PASSED TO THE      UNCHANGED                  *
*              ROUTINE BY JES2:                                     *
*                0  INDICATES JECL                                  *
*                4  INDICATES JCL                                   *
*    R1        ADDRESS OF 3-WORD         UNCHANGED                  *
*              PARAMETER LIST:                                      *
*               +0  ADDRESS OF IMAGE BUFFER                         *
*               +4  ADDRESS OF RDWFLAGX                             *
*               +8  ADDRESS OF JCTXWRK                              *
*    R2-R9     N/A                       UNCHANGED                  *
*    R10       ADDRESS OF THE JCT OR ZERO UNCHANGED                 *
*    R11       ADDRESS OF THE HCT        UNCHANGED                  *
*    R12       N/A                       UNCHANGED                  *
*    R13       ADDRESS OF THE PCE        UNCHANGED                  *
*    R14       RETURN ADDRESS            UNCHANGED                  *
*    R15       ENTRY ADDRESS             RETURN CODE                *
*                                                                   *
* RETURN CODES (R15 ON EXIT)                                        *
*                                                                   *
*       0  TELLS JES2 THAT IF THERE ARE ANY ADDITIONAL EXIT ROUTINES *
*          ASSOCIATED WITH THIS EXIT, CALL THE NEXT CONSECUTIVE     *
*          EXIT ROUTINE.  IF THERE ARE NO OTHER EXIT ROUTINES ASSO- *
*          CIATED WITH THIS EXIT, CONTINUE WITH NORMAL PROCESSING.  *
*                                                                   *
*       4  TELLS JES2 THAT EVEN IF THERE ARE ADDITIONAL EXIT ROUTINES *
```

```
*          ASSOCIATED WITH THIS EXIT, IGNORE THEM.  CONTINUE WITH     *
*          NORMAL PROCESSING.                                         *
*                                                                     *
*       8  FOR JES2 CONTROL STATEMENTS, TELLS JES2 NOT TO PERFORM     *
*          STANDARD HASPRCCS PROCESSING; INSTEAD, IMMEDIATELY         *
*          CONVERT THE STATEMENT TO A COMMENT (//*) WITH THE NULL-    *
*          ON-INPUT FLAG SET TO ONE AND WRITE THE STATEMENT TO THE    *
*          JCL DATA SET.  FOR JCL STATEMENTS, TELLS JES2 TO PERFORM   *
*          STANDARD HASPRDR PROCESSING.                               *
*                                                                     *
*      12  TELLS JES2 TO CANCEL THE JOB BECAUSE AN ILLEGAL CONTROL    *
*          STATEMENT HAS BEEN DETECTED; OUTPUT IS PRODUCED.           *
*                                                                     *
*      16  TELLS JES2 TO PURGE THE JOB; NO OUTPUT IS PRODUCED.        *
*                                                                     *
* MACROS = JES2 - $ENTRY, $MODEND, $MODULE, $MSG, $RETURN, $SAVE,     *
*                 $WTO                                                *
*                                                                     *
* MACROS = MVS  - NONE                                                *
*                                                                     *
* CHANGE ACTIVITY:                                                    *
*                                                                     *
***********************************************************************
*
*        TITLE 'JES2 USER EXIT 04-- PROLOG ($HASPGBL)'
         COPY  $HASPGBL
*
JES2X04  $MODULE ENVIRON=JES2,                                       C
               $BUFFER,          GENERATE HASP I/O BUFFER DSECT      C
               $CAT,             GENERATE HASP CAT DSECT             C
               $CMB,             GENERATE HASP CMB DSECT             C
               $DCT,             GENERATE HASP DCT DSECT             C
               $DTE,             GENERATE HASP DTE DSECT             C
               $ERA,             GENERATE HASP ERA DSECT             C
               $HASPEQU,         GENERATE HASP EQUATES DSECT         C
               $HCT,             GENERATE HASP HCT DSECT             C
               $JCT,             GENERATE HASP JCT DSECT             C
               $JQE,             GENERATE HASP JQE DSECT             C
               $KIT,             GENERATE HASP KIT DSECT             C
               $MIT,             GENERATE HASP MIT DSECT             C
               $PADDR,           GENERATE HASP PADDR DSECT           C
               $PCE,             GENERATE HASP PCE DSECT             C
               $PIT,             GENERATE HASP PIT DSECT             C
               $RDRWORK,         GENERATE HASP RDRWORK DSECT         C
               $TQE,             GENERATE HASP TQE DSECT             C
               $USERCBS,         GENERATE HASP USERCBS DSECT         C
               $XECB,            GENERATE HASP XECB DSECT            C
               $XIT,             GENERATE HASP XIT DSECT             C
               ASCB,             GENERATE MVS ASCB DSECT             C
               RPL               GENERATE MVS RPL DSECT
JES2X04  RMODE ANY
*        TITLE 'JES2 USER EXIT 04--- JCL/JECL SCAN'
***********************************************************************
* REGISTER USAGE (INTERNAL)                                           *
*                                                                     *
*    REG       VALUE                                                  *
*                                                                     *
*    R0        PARAMETER FROM JES2                                    *
*    R1-R9     WORK REGISTERS                                         *
*    R10       JCT ADDRESSABILITY                                     *
*    R11       HCT ADDRESSABILITY                                     *
*    R12       EXIT4 ADDRESSABILITY                                   *
*    R13       PCE ADDRESSABILITY                                     *
```

```
*    R14       LINK/WORK REGISTER                                    *
*    R15       LINK/WORK REGISTER                                    *
*                                                                    *
**********************************************************************
         SPACE 1
EXIT4   $ENTRY  BASE=(R12)         PROVIDE EXIT ROUTINE ENTRY POINT
        $SAVE                      SAVE CALLER'S REGISTERS
         USING JCT,R10             ESTABLISH JCT ADDRESSABILITY
         USING HCT,R11             ESTABLISH HCT ADDRESSABILITY
         SPACE 1
         LR    R6,R1               SAVE POINTER TO PARAMETERS
         LR    R7,R0               SAVE FLAG PARAMETER
         LR    R12,R15             ESTABLISH BASE REGISTER
         LTR   R10,R10             IF JCT NOT PRESENT
         BZ    X4RC00              IF JCT=0, NOT JCL
*
         CLI   JCTJOBID,C'J'       NOT INTERESTED IN STC OR TSO USER
         BNE   X4RC00                  SO LEAVE FOR THESE
         LTR   R7,R7               R0=0 ON ENTRY IMPLIES CALL IS FOR A
         BNZ   X4RC00                  JES2 CONTROL STMNT
* ------------------------------------------------------------------
*  IF THIS IS A '/*ROUTE' OR '/*XEQ' CARD, PROCESSING TO FIND THE
*  TARGET SYSTEM ID WILL BE COMMON.  JUST HAVE TO CHECK FIRST THAT
*  THE ROUTE CARD IS NOT FOR PRINT.
* ------------------------------------------------------------------
JESCC    DS    0H
         CLI   JCTUSER3,X'FF'      HAVE WE SEEN A SYSAFF YET ?
         BE    X4RC00              YES. SKIP FURTHER PROCESSING
*
         L     R3,0(0,R6)           GET ADDRESS OF JCL CARD
         CLC   =CL8'/*ROUTE ',0(R3) IS IT A ROUTE CARD?
         BE    LBL0012              - YES, PROCESS
         CLC   =CL6'/*XEQ ',0(R3)   IS IT AN XEQ CARD?
         BNE   X4RC00               - NO, BRANCH
         LA    R1,5(R3)             POINT R1 PAST THE XEQ
         B     LBL0015              AND CHECK FOR TARGET SYSID
* ------------------------------------------------------------------
*  CHECK HERE TO ENSURE THAT THE '/*ROUTE' CARD IS FOR AN XEQ AND
*  NOT A PRINT.
* ------------------------------------------------------------------
LBL0012  DS    0H
         LA    R1,7(R3)             POINT PAST THE '/*ROUTE'
         TRT   0(72-7-8,R1),TBNBLNK FIND FIRST NON-BLANK
         BZ    X4RC00               REST OF CARD BLANK SO LEAVE
         CLC   =CL3'XEQ',0(R1)      FOUND AN XEQ?
         BNE   X4RC00               - NO, LEAVE
         LA    R1,3(R1)             POINT PAST THE 'XEQ'
* ------------------------------------------------------------------
*  COMMON PROCESSING TO IDENTIFY THE TARGET SYSTEM ID.
*  - AFTER THE EXECUTE, R1 POINTS TO THE TARGET ID
* ------------------------------------------------------------------
LBL0015  DS    0H
         LA    R15,72-1(,R3)
         SR    R15,R1                  USE TRT 0(*,*,R1),TBNBLNK
         EX    R15,TRTNBLNK            TO FIND NEXT NON-BLANK
         BZ    X4RC00                  - DIDN'T FIND ONE SO LEAVE
*
         MVC   JCTUSER3,BLANKS         CLEAR JCTUSER3
         MVC   JCTUSER4,BLANKS         CLEAR JCTUSER4
LBL0020  DS    0H
         LA    R2,8                    NAME NAME LENGTH MAXIMUM
         LA    R3,JCTUSER3             ADDR TO PUT NODE NAME
LBL0030  DS    0H
```

```
         CLI   0(R1),C' '              END OF NODE NAME ?
         BE    LBL0040                 YES. BRANCH
         CLI   0(R1),C','              END OF NODE NAME ?
         BE    LBL0040                 YES. BRANCH
         MVC   0(1,R3),0(R1)           MOVE A NODE NAME CHARACTER
         LA    R1,1(0,R1)              MOVE TO NEXT CHARACTER
         LA    R3,1(0,R3)              MOVE TO NEXT CHARACTER
         BCT   R2,LBL0030              LOOP THRU NODE NAME
* ------------------------------------------------------------------*
*  ADD A /*JOBPARM SYSAFF=XXX IF JOB DESTINED FOR NEW SYSPLEX SYSTEM *
* ------------------------------------------------------------------*
LBL0050  DS    0H
         MVC   JCTXWRK(80),AFFCARD     BUILD SYSTEM AFFINITY STMT
         MVC   JCTXWRK+17(8),JCTUSER3  ADD SYSTEM NAME
         OI    RDWFLAGX,RDWXXSNC       INDICATE STATEMENT INSERTED
         B     X4RC00
*
AFFCARD  DC    CL80'/*JOBPARM SYSAFF='
*
* ---------------------------------------------------------------
*   COMMON EXIT
* ---------------------------------------------------------------
X4RC00   DS    0H
         LA    R15,0             SET RC=0
         $RETURN RC=(R15)        SAVE RETURN CODE
        SPACE 3
       EJECT
* ---------------------------------------------------------------
*   LITERALS
* ---------------------------------------------------------------
BLANKS   DC    CL4' '
* ---------------------------------------------------------------
*   NON-BLANK CHECKING
* ---------------------------------------------------------------
TRTNBLNK TRT   0(*-*,R1),TBNBLNK         ** EXECUTED
*
TBNBLNK  DC    256X'FF'
         ORG   TBNBLNK+C' '
         DC    X'00'
         ORG   ,
*
        LTORG
        $MODEND
        END
$$
```

22

Maintenance considerations

This chapter discusses the considerations for rolling out new releases, and planned maintenance for a system that is being moved into an existing sysplex.

22.1 Controlling service, releases, and propagation

There are many things to consider when developing a maintenance strategy, setting up the structure of a maintenance environment, and propagating service and releases to systems in a sysplex. In this chapter, we identify and suggest ways of dealing with these considerations for customers wanting to move into a BronzePlex, a GoldPlex or a PlatinumPlex configuration.

Throughout this chapter, we use the terms listed in Table 22-1.

Table 22-1 Maintenance definitions

Term	Definition
Maintenance environment	A set of SMP/E target and distribution libraries and CSIs, or non-SMP/E product target libraries, that are used to maintain the operating system, subsystems, and related program products. These libraries are the source for the system runtime environment, but would never actually be IPLed from.
sysres or sysres set	Either a single volume, or a group of volumes, utilizing the multi-volume sysres support. Due to the amount of software included with z/OS, even small z/OS systems now require more than a single sysres volume.
Version HFS	The IBM-supplied root HFS data set containing files and executables for the operating system elements. We use this term to avoid confusion with the sysplex root HFS data set.

While we recommend that you merge the maintenance environments when moving a system into a sysplex, there are no technical requirements for doing this. You need to analyze your requirements to decide the structure that best fits your environment's needs. To help you understand your requirements, review the considerations in Table 22-2 on page 367.

In this chapter, we specifically address the z/OS environment; however, the principles could easily be applied to the subsystems as well. While you could merge all the environments (z/OS, CICS, DB2, and so on) into a single SMP/E environment, we recommend maintaining a separate SMP/E environment for each SREL, as this provides more flexibility and separation of responsibilities. Also, the environment that this discussion is based on is as follows:

- There is one SMP/E Global zone, which is used for all z/OS systems.
- The libraries pointed to by the Target zone DDDEFs are normally not used to IPL. Those libraries are used to run SMP/E Apply jobs against, and they are then copied to create IPLable sysreses.
- In a BronzePlex, the libraries would be copied to tape, and then back to the target sysres. In a GoldPlex or a PlatinumPlex, they would be copied directly to the target sysres.
- All SMP/E jobs are normally run on a specific, non-production, system.

It is important to have, or develop, a maintenance strategy that meets your technical and business requirements. A preventative maintenance strategy can help avoid system or subsystem outages—a large percentage of system outages experienced by customers are caused by problems that already have a fix available for more than six months.

However, when merging a system into a sysplex, you also need to consider the usage of each system, in terms of the maintenance practices and strategy. Refer to 22.4, "Maintenance strategy" on page 371 for more information.

In line with the target sysplex environment options outlined in 1.2, “Starting and ending points” on page 2, moving the incoming system into a BronzePlex would typically be done to obtain the benefits of PSLC or WLC, so there is no requirement to merge the maintenance environments of the incoming system into the target sysplex. This would mean maintaining the two maintenance environments and sysres structures as you do today.

We would, however, recommend that some merging or consolidation be considered. Maintaining multiple maintenance environments increases system programmer workload and increases the chances of mistakes being made. If possible, we recommend merging into a single SMP/E Global zone so Enhanced HOLDDATA would only need to be received once, and all SMP/E management can be done through a single interface. Of course, limited DASD sharing and catalog issues in the BronzePlex configuration may make this difficult to achieve.

Moving the incoming system into a GoldPlex would be done to share the system software environment (sysres), plus other selected shared system infrastructure, so merging the maintenance environment would be essential. “Merging”, in this case, means understanding the product mix between systems and making all required products available on a single sysres set.

A GoldPlex may or may not include the sharing of the Master Catalog between all systems in the target sysplex, so the user catalogs may or may not also be shared. Therefore, you will need to consider how you want to manage the sysres and the maintenance environment’s data set catalog entries. Assuming most, if not all, data sets on the sysres are cataloged in the Master Catalog, you will need to define a process to keep the Master Catalogs in sync across the sysplex.

You will also need to consider how you will manage data sets such as distribution libraries and SMP/E data sets within the maintenance environment. Either all those data sets must be in a user catalog that is shared by all systems in the sysplex, or else all management of the maintenance environment must be done from a single system.

Moving the incoming system into a PlatinumPlex would result in shared DASD, a shared sysres, maintenance environment, Master Catalog and all user catalogs, a single SMSplex, and so on. This configuration allows you to avail of the benefits of full sharing, and the maintenance environment data sets will be available from any system within the sysplex.

22.2 Considerations when merging systems into a sysplex

When moving the incoming system into a target sysplex, the maintenance considerations will depend on the features you plan to exploit within the sysplex. To help understand the requirements, review the considerations in Table 22-2.

Table 22-2 Maintenance considerations when merging systems into a sysplex

Consideration	Note	Type	Done?
Ensure the incoming system is within the coexistence window.	**1**	**B**	
Review the coexistence maintenance requirements and obtain and apply any required service.	**1**	**B, G, P**	
Review the maintenance policy for the incoming system and target sysplex.	**2**	**G, P**	
Review the product mix of the incoming system and target sysplex.	**3**	**G, P**	
Ensure sysres data sets are cataloged across the sysplex.	**4**	**G**	

Consideration	Note	Type	Done?
Choose a HLQ for maintenance and SMP/E data sets that will allow for where those data sets will be accessed from.	**4**	**G**	
Merge into a single global zone.	**5**	**B, G, P**	
Maintenance environment data set placement.	**5**	**G, P**	
Review HFS data set placement.	**6**	**G, P**	
Review naming conventions for the data sets on the sysres volumes.	**7**	**G, P**	
Ensure propagation routine checks for active sysres volumes within the sysplex.	**8**	**G, P**	

The "Type" specified in Table 22-2 on page 367 relates to the sysplex target environment—**B** represents a BronzePlex, **G** represents a GoldPlex, and **P** represents a PlatinumPlex. The notes highlighted in Table 22-2 refer to the following points:

1. This consideration only applies to environments where the incoming system and the target sysplex systems will not be sharing a sysres, which is most likely a BronzePlex.

   You will need to ensure that the operating system of the incoming system is at the appropriate level to coexist with the other systems within the target sysplex. You will also need to ensure that the appropriate coexistence maintenance is applied to all systems within the target sysplex prior to merging the incoming system into the target sysplex; refer to 22.3, "Coexistence levels" on page 370 for more information. For additional information on the options available to you for obtaining the required maintenance, refer to 22.4.4, "How to obtain the required maintenance" on page 375.

2. It is important to have, or develop, a maintenance strategy that matches your technical and business requirements. A preventative maintenance strategy can help avoid system or subsystem outages. However, when merging a system into a sysplex, you also need to consider the usage of each system (for example, production systems are typically updated less frequently than test systems), in terms of the maintenance practices and strategy. Refer to 22.4, "Maintenance strategy" on page 371 for more information.

3. If you are planning to use a single sysres, you will need to review the product mix to ensure all the products available on the sysres currently being used by the incoming system are also available on the shared sysres.

4. The GoldPlex configuration might not include the sharing of the Master Catalog. Equally, you may or may not plan on sharing the system-related user catalogs (for example, those containing the distribution libraries). Therefore, you will need to consider how you want to manage the sysres data set catalog entries. Assuming most, if not all, data sets on the sysres are cataloged in the Master Catalog you will need to define a process to keep the Master Catalogs in sync across the sysplex.

   You will also need to consider how you want to manage the maintenance environment's data set catalog entries. The options would be to either maintain your maintenance environment from a single system within the target sysplex, or make the HLQ(s) for the data sets within the maintenance environment, such as the distribution libraries and SMP/E data sets, available in a user catalog that is shared by all systems within the sysplex. If you choose the latter option, there may be issues if your maintenance environment's data sets are SMS-managed. In a PlatinumPlex configuration, there would be a single Master Catalog and a single SMSplex, so this would not be an issue.

   We would recommend that, in the GoldPlex configuration, you maintain your maintenance environment from a single system within the target sysplex. For suggested data set naming standards, refer to Table 22-3 on page 380.

5. A shared maintenance environment reduces system programmer workload. If possible, we recommend merging into a single OS/390 or z/OS SMP/E Global zone so Enhanced HOLDDATA would only need to be received once; refer to 22.4.5, “Enhanced HOLDDATA” on page 377 for more information. Limited DASD sharing and catalog issues may make this impossible to achieve in a BronzePlex configuration, but it should be achievable in a GoldPlex or PlatinumPlex configuration.

   You can use the SMP/E GZONEMERGE command to merge the Global zones from the incoming system into an existing Global zone on the target sysplex. Refer to 22.6.1, “Merge global zones” on page 392 for more information.

   A single maintenance environment and shared sysres structure within the target sysplex would provide a single logical system image running numerous systems. Refer to 22.5, “The maintenance environment” on page 379, for information about a suggested structure for the maintenance environment.

6. A copy of the Version HFS is required for the maintenance environment, as well as each of the sysres sets. The placement of the HFS is flexible; it could be on or off the sysres volumes, and SMS- or non-SMS-managed.

   We recommend placing the HFS on one of the sysres volumes for ease of management, especially in a BronzePlex or a GoldPlex configuration, because of the limited DASD sharing and separate SMSplexes. For more information, refer to **22.5.2, “HFS considerations” on page 383**.

7. You will need to consider the data set naming standards of the incoming system and the target sysplex if you are planning to share the sysres. If there are differences in any of the system data set names on the sysres volumes, there may be problems with data set allocation in JCL, PROCs, execs, and so on. For example, if the incoming system has a naming standard of SORT.SORTLIB and the target sysplex uses SYS1.SORTLIB, then there may be JCL, PROCs, and so on that have this data set name hardcoded in STEPLIBs and need to be changed prior to, or during, the implementation of a shared sysres.

   An alternative, although probably not desirable in the long term because of the maintenance overhead, is to set up an alias for SORT.SORTLIB pointing to SYS1.SORTLIB. This would ensure that anyone using the standard catalog search order to find SORT.SORTLIB would be directed to SYS1.SORTLIB, unaware that the actual data set name had changed.

   Consider implementing system symbols on the incoming system to eliminate any hardcoded data set names prior to merging into the target sysplex. System symbols can be used in most of the Parmlib members, system commands, JCL for started tasks and TSO/E logon procedures, JES2 initialization statements and commands, JES3 commands, dynamic allocations, and others. They *cannot*, however, be used in batch job JCL. For more information on system symbols, refer to 10.2, “System symbols” on page 144.

8. An important requirement of any cloning or propagation process, in a shared sysres environment, is to ensure it does not try to overwrite a sysres that is active within the target sysplex. Obviously, checking the target sysres prior to running the propagation routine is required, but mistakes can happen. One way of doing this programmatically is outlined in Example 22-8 on page 386.

22.3 Coexistence levels

IBM has a maintenance policy that, at the time of writing, allows the coexistence of up to four concurrent operating system releases within the same sysplex. IBM uses “n” to represent the current release. Due to a special provision, OS/390 R10 and z/OS V1R1 are treated as a single coexistence level rather than two levels because of the unique relationship between OS/390 R10 and z/OS V1R1. Therefore, for example, you could have systems from OS/390 2.10 to z/OS 1.4 coexisting within the target sysplex, as shown in Figure 22-1 on page 370.

Note: As part the announcement for z/OS 1.4, IBM has announced changes to the maintenance and coexistence strategy for z/OS, bringing the two strategies into line with each other. As a result, in future releases, each release will be supported for three years from its general availability date, and up to three concurrent releases of z/OS will be able to co-exist within the sysplex. This change will start to take effect with z/OS 1.4. For more details, refer to the following site:

http://publibz.boulder.ibm.com/cgi-bin/bookmgr_OS390/BOOKS/EOZ2B131/5.3?DT=20021003160915

Coexistence occurs when two or more systems at different software levels share resources. The resources could be shared at the same time by different systems in a multisystem configuration. Examples of coexistence are: two different JES releases sharing a spool; two different service levels of DFSMSdfp™ sharing catalogs; multiple levels of SMP/E processing SYSMODs packaged to exploit the latest enhancements; or an older level of the system using the system control files of a newer level. The sharing of resources is inherent in multisystem Parallel Sysplex configurations. The way in which you make it possible for systems to coexist within a sysplex environment is to install coexistence service (PTFs) on the earlier-level systems.

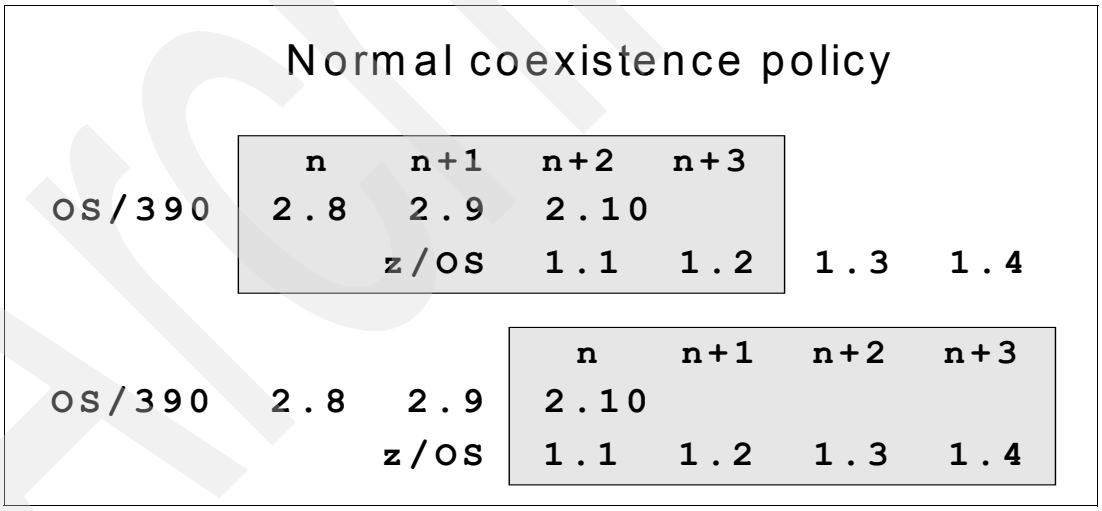


Figure 22-1 Operating system coexistence policy

You will need to ensure that the operating system of the incoming system is at the appropriate level to coexist with the other systems within the target sysplex. You will also need to ensure that the appropriate coexistence maintenance is applied to all systems within the target sysplex and the incoming system, prior to bringing the incoming system into the target sysplex.

Subsystem levels will also need to be reviewed to ensure there are no compatibility, coexistence, or toleration maintenance requirements. This is a requirement if either the incoming system or target sysplex are to be used as a target for subsystem restarts using ARM, or a similar automatic restart mechanism, and either are at a different operating system level than the other.

To obtain a list of the required service, refer to Chapter 5, “Ensuring coexistence and fallback”, in *OS/390 Planning for Installation,* GC28-1726, or *z/OS Planning for Installation,* GA22-7504, at the appropriate operating system level.

22.4 Maintenance strategy

It is important to have a maintenance strategy that suits your technical and business requirements. A preventative maintenance strategy can help avoid system or subsystem outages. However, when merging a system into a sysplex, you need to consider the usage of each system, in terms of the maintenance practices and strategy.

For example, if one or more of the systems within the target sysplex are production IMS systems and the incoming system is a WebSphere development system, the maintenance strategy may differ and your practices may need to be refined accordingly. It is not possible to define a single maintenance approach that will work for all sites; therefore, in 22.5, “The maintenance environment” on page 379, we provide an example of a maintenance structure that could be used to provide the flexibility to meet both types of maintenance requirements.

Your maintenance strategy should be based on availability requirements. Staying current on maintenance can avoid outages caused by known defects. Your preventive maintenance plan for improved availability in a sysplex environment should be well defined, and maintenance should be installed regularly. For information on the latest IBM recommendations and tools to help with this task, refer to the IBM Redpaper entitled *Improvements in z/OS Service*, REDP0324, available on the Redbooks Web site at:

`http://www.redbooks.ibm.com`

22.4.1 Maintenance options

IBM provides service for current S/390 and z/Series products, including OS/390 and z/OS. PTFs are available for corrective and preventive maintenance. The most common service deliverables for obtaining maintenance are:

- ShopzSeries

  ShopzSeries allows you to order both corrective and preventive service by using the Internet. z/OS maintenance, both corrective and preventive, can be delivered through the Internet or by using standard physical media delivery. IBM is deploying this service around the world, and it is planned to be available in all geographies by the end of 2003. Check with your local IBM representative or the ShopzSeries Web site for availability. See the ShopzSeries section of 22.4.4, “How to obtain the required maintenance” on page 375 for more information.

- Enhanced Service Offering (ESO)

  ESO is available in two flavors—a subscription service, and on-request. The ESO subscription service provides monthly program error (PER)-closed PTFs. ESOs are shipped at intervals you specify (from 1 to 12 months), and will contain the PTFs based on the selection criteria you specify. ESOs are shipped approximately once a month and will contain PTFs that have been available as corrective service for at least 90 days.

The on-request option is only shipped when you specifically request it, and it provides *all* PTFs, not just those that are available for 90 days or more.

- Customer Built Product Delivery Option (CBPDO)

  CBPDO provides monthly PER-closed PTFs and, in addition, includes corrective (COR) closed reach-ahead fixes for HIPERS and PEs on a weekly basis. CBPDOs are normally ordered on an as-needed basis; however, some countries provide the option of delivering CBPDOs on a subscription basis.

- ServerPac Offering

  The ServerPac delivery vehicle integrates service not only for z/OS itself, but for all orderable products on the z/OS system replace checklist. The corrective service follows the traditional distribution procedures.

- RefreshPac Offering

  RefreshPac is a fee offering that provides tailored service based on a copy of the SMP/E CSIs that you send to IBM. The service shipped in the RefreshPac is install-tested in IBM prior to shipment to ensure that it will SMP APPLY cleanly.

  When you order a RefreshPac/MVS, you can optionally order the automatic shipment of periodic follow-on service packages for your order. These packages, called “selective follow-on service”, contain service that is applicable to your RefreshPac/MVS order and that has become available since the latest preceding order or selective follow-on service package was built. When you request selective follow-on service, you specify the desired number of follow-on packages (1 or 2) and the number of days between packages (21 to 60).

- You can also obtain individual or groups of PTFs using the SRD function in IBMLINK. Requested PTFs will be either shipped over the network or on tape media.

22.4.2 Consolidated Service Test (CST)

In the past, many of the key components on the z/OS software platform had different recommended maintenance strategies, with little or no coordination between them. This had the potential to cause confusion, since it was sometimes unclear what level of service each product or subsystem was recommending.

In an effort to eliminate this confusion, IBM established the Consolidated Service Test (CST) team. This cross-product team’s mission is to enhance the way IBM tests software and to come up with one consistent recommendation for maintenance for z/OS software, including the major subsystems. The CST team consists of product test representatives from the operating system and the major subsystems, including z/OS, OS/390, CICS, DB2, IMS, and MQSeries. The goals are to improve the overall quality of service, and provide a common maintenance recommendation for all products on the platform.

Products and levels tested

CST testing is in addition to existing testing programs and does not replace any current testing performed by the products. CST is currently focused on the Parallel Sysplex environment. It is a customer-like production sysplex environment in an IBM test lab with batch and data-sharing applications that exploit and stress the latest functions with up to two levels of subsystems on three levels of z/OS or OS/390 systems. The key products specifically tested in CST include:

- z/OS
- OS/390
- CICS Transaction Server for OS/390
- DB2 UDB for OS/390

- IMS
- MQSeries for OS/390

The WebSphere environment is being considered as a future addition.

For more information on CST, visit the CST Web site:

http://www.ibm.com/servers/eserver/zseries/zos/servicetst/

22.4.3 Recommended Service Upgrade (RSU)

Recommended Service Upgrade (RSU) has been available, with preventive maintenance deliverables such as ESO and CBPDO, on the z/OS platform since 1996. It is the subset of all PTFs that are recommended for installation by all customers. RSU is provided as a file on the service deliverables in the form of SMP/E ++ASSIGN statements for RSUyymm. RSU maintenance has been tested monthly and is recommended as the preventive service strategy for z/OS elements.

Although RSU is available for all products on the platform, prior to the creation of the CST process, not all products made the same preventive service recommendation. The recommendation has varied by product. For example, z/OS recommended using RSUs as the basis for selecting service to be applied, CICS recommended that all maintenance be installed, and DB2 created their own list of recommended service called Recommended Maintenance Lists (RMLs).

The criteria for inclusion in the original RSU was:

- Severity 1 and 2 APARs
- HIPERs APARs
- Special Attention APARs
- Fixes for PEs
- Security/Integrity APARs

Revised RSU

Beginning in November 2001, IBM changed its recommendation for service and redefined the criteria for how RSUs are assigned to fixes. This new recommendation is complemented by the additional testing performed by CST. Like the prior RSU process, recommendations come out monthly in the form of SMP/E ++ASSIGN statements (RSUyymm). You can order both the RSU ++ASSIGN statements, and the PTFs associated with a given RSU using SUF and ShopzSeries; see 22.4.4, “How to obtain the required maintenance” on page 375 for more details on ShopzSeries and SUF.

However, the criteria for deciding which RSU should be assigned to a PTF has undergone a number of changes. The biggest change is that a PTF must have gone through a CST monthly test cycle prior to being assigned an RSU SOURCEID. This additional test cycle allows IBM to identify and eliminate problems that may occur when the products are integrated together. This change, in when the recommendation is made, affects when the PTFs are assigned an RSU SOURCEID. The RSU SOURCEID no longer reflects when the PTF closed. Rather, it now reflects when the service completed the test cycle and became recommended. The PUTyymm SOURCEID can be used to identify when a PTF closes.

Keep in mind that the date that a PTF is marked recommended does not affect when that PTF is available for corrective service—that remains unchanged. There is also no change to when SOURCEIDs, such as HIPER or PRP (PTFs that resolve PE’d PTFs), are assigned to the PTFs.

The next change to assignment of RSU SOURCEIDs is that it is now based on a different criteria. Each quarter, *all* service (Severity 1, 2, 3 and 4 APARs) as of the end of the prior quarter will have completed a CST test cycle, and will therefore become recommended. Additionally, each *month,* HIPER PTFs, PTFs that resolve PE PTFs, and other fixes as warranted that have completed a CST test cycle will become recommended.

Both the quarterly and the monthly recommendations use the same RSUyymm SOURCEID notations, so you can identify the quarterly recommendations by their month values (for example, RSUyy03, RSUyy06, RSUyy09, and RSUyy12). As always, you should review HIPERs and PE fixes on a regular basis and install those applicable to your environment.

The benefits of the revised RSU process are:

- Better testing of maintenance through a coordinated effort by CST and representatives from each of the participating products in a Parallel Sysplex environment.
- The recommendation for inclusion in an RSU is made by the participating product experts.
- Testing is completed prior to the RSU being made available. The original RSU was based on the selection criteria and the testing was done after the RSU was generally available.
- The new process allows for consistent maintenance recommendations across participating products (both in what service is recommended, and how frequently preventive maintenance should be applied).

The maintenance included in each RSU differs, depending on whether it is a monthly or quarterly RSU.

Quarterly Consolidated Service Test cycles

CST has a quarterly testing cycle. Each quarter begins by installing all PTFs that were closed in the prior quarter, and defining test scenarios to exploit new product functions while existing workloads are being executed.

During the second month of the quarter, CST runs the new test scenarios, identifies any problems, and applies fixes as necessary. In the final month of the quarter, recovery tests are performed and workloads are run in a high stress environment. At the end of the quarter, results are published in the Consolidated Service Test report.

Quarterly RSUs

RSUyy03, RSUyy06, RSUyy09, and RSUyy12 contain:

- All service, Severity 1, 2, 3, and 4 APARs, that were successfully tested by CST as of the end of the prior quarter.
- HIPERs, PE fixes, and security/integrity APARs that have completed a 30-day CST cycle in that month.
- CST corrective fixes.

Monthly Consolidated Service Test cycles

At the end of each month, between quarterly recommendations, CST will provide a delta recommendation for customers whose preventive strategy requires more frequent maintenance updates, or whose upgrade windows don't align with quarterly boundaries.

The monthly recommendation supports the most recently published CST quarterly recommendation. It also includes HIPERs, PE fixes, and security/integrity fixes that have been installed at the beginning of the month and tested for the duration of the month, plus fixes to any problems found in CST testing.

Monthly RSUs

RSUyy01, RSUyy02, RSUyy04, RSUyy05, RSUyy07, RSUyy08, RSUyy 10, and RSUyy11 contain:

- HIPERs, PE fixes, and security/integrity APARs that have completed a 30-day CST cycle in that month.
- CST corrective fixes.

22.4.4 How to obtain the required maintenance

You have a number of alternatives for ordering and taking delivery of corrective and preventative maintenance. The following is a brief outline of four of those options:

- ShopzSeries

  ShopzSeries allows you to order both corrective and preventive service by using the Internet. z/OS maintenance, both corrective and preventive, can be delivered through the Internet, or by using standard physical media delivery.

  Obtaining service using ShopzSeries reduces your research time and effort by using your uploaded SMP/E Consolidated Software Inventory (CSI) to ensure that all applicable service—(including reach-ahead service) for the installed FMIDs in the target zones—is selected. ShopzSeries can be used in place of the S/390 Service Update Facility (SUF) tool.

  IBM is deploying this service around the world and plans to have it available in all geographies by the end of 2003. Check with your local IBM representative or the ShopzSeries Web site for availability.

  ShopzSeries allows you to provide an installed service inventory that can be used to ensure that you are only sent service that has not already been RECEIVEd or APPLYed, and which includes all applicable requisites, should you choose to include requisites. All packages include Enhanced HOLDDATA.

  ShopzSeries allows you to:

  - Request PTFs either by PTF number or by APAR number.
  - Order preventative service for all licensed products in an SREL (like a service-only CBPDO).
  - Request all service (like an ESO or CBPDO) based on a service inventory.
  - Request service for the latest RSU.
  - Request just HIPER- and PE-fixing PTFs.
  - Get the package transported via either:
    - SMP/E RECEIVE FROMNETWORK
    - HTTP, using a store and forward model.
    - FTP, either to a host or a store and forward model.
  - Receive a compressed package (which used GIMZIP rather than TERSE) prior to transport.
  - Receive a package up to 1 GB (compressed) in size.

  For corrective service, you specify a list of either PTFs or APARs, and can optionally run an SMP/E job which creates an installed service inventory of your SMP/E zones (which is a simple way of saying a history of all the FMIDs and PTFs that have been RECEIVED or APPLYed in the SMP/E target zones you specify).

  If you provide a service inventory, IBM then works out what other PTFs are required to install your requested service, including *all* requisite service and resolution of any PE chains, filtering out any service already RECEIVEd or APPLYed.

If you do not provide a service inventory, you will receive the requested service without any additional requisites. The service (and two years of Enhanced HOLDDATA) is then either available to be downloaded electronically, or sent physically on 3480, 3490E, or 3590 cartridges.

For preventive service, you run the same SMP/E job to create an installed service inventory of your SMP/E zones. IBM then determines what service is required to bring your system to the latest RSU level, install the latest HIPER or PRP service, or install all available service. IBM will then create a package for you containing the appropriate maintenance. The package will eliminate all PTFs that you have already RECEIVEd, as well as including any PTFs that IBM determined you need as requisites. It will also contain resolution of known PEs. Since your service order is based on your target SMP/E environment, it can be delivered quickly because it will not include any extraneous service.

Tracking the status of your order

For either corrective service or preventive service, you can track the status of your order until it is ready for you to pull down to your host or workstation. You can either use z/OS 1.2 SMP/E (also available as SMP/E for z/OS and OS/390 V3R1, or as a Web download) to RECEIVE the service directly from the Internet using the SMP/E RECEIVE FROMNETWORK command.

Alternatively, you can FTP it to your mainframe and install it with the normal SMP/E RECEIVE statement, or you can download it to your workstation. Once you have the order on your workstation, you can upload it to your host for processing. Of course, physical delivery of your order is also an option, if you can't take electronic delivery.

In addition, all orders larger than 1 GB compressed will be shipped on the physical media of your choice, or not at all, if you decide you'd rather resubmit your order to request less service.

For more information about ShopzSeries, refer to the following URL:

`https://www14.software.ibm.com/webapp/ShopzSeries/ShopzSeries.jsp`

- TechSupport

  TechSupport allows you to order corrective service by using the Internet. z/OS maintenance can be delivered through the Internet, or by using standard physical media delivery.

  TechSupport allows you to:

  - Order PTFs by providing a list of PTFs that you want. (It does *not* support ordering by APAR numbers.)
  - Get the package transported via FTP, which is TERSEd prior to transport.

  The TechSupport Web site is available at the following URL:

  `https://techsupport.services.ibm.com/server/login`

- Service Request & Delivery (SRD)

  SRD is an option within IBMLINK that allows you to order both corrective and preventive service using the Internet. This service is available via subscription. z/OS maintenance, both corrective and preventive, can be delivered through the Internet or by using standard physical media delivery. SRD also supports VM.

  SRD allows you to provide an installed service inventory which can be used to ensure that you are only sent service that has not already been RECEIVEd or APPLYd, and includes all applicable requisites, if you choose to include requisites.

SRD allows you to:

- Order PTFs by providing a list of PTFs that you want (it does *not* support ordering by APAR number).
- Request just HIPER- and PE-fixing PTFs based on a service inventory profile.
- Request preventive maintenance based on a service inventory profile.
- Request an ESO.
- Get the package transported via FTP, which is TERSEd prior to transport.

SRD is available via IBMLINK at the following URL:

`http://www.ibmlink.ibm.com/`

- Service Update Facility (SUF)

  SUF allows you to order both corrective and preventive service by using the Internet. OS/390 or z/OS maintenance, both corrective and preventive, can be delivered through the Internet or by using standard physical media delivery. SUF also supports VM and VSE.

  SUF allows you to provide an installed service inventory, which can be used to ensure you are only sent service that hasn't already been RECEIVEd or APPLYed, and which includes all applicable requisites, if you choose to include requisites.

  SUF allows you to:

  - Request PTFs by either PTF number or APAR number.
  - Request service by RSU.
  - Request just HIPER- and PE-fixing PTFs.
  - Request Enhanced HOLDDATA.
  - Get the package transported via FTP, which is TERSEd prior to transport.

  The SUF Web site contains a “test drive” facility, so you can see how SUF works and the types of functions it delivers. More information on SUF is available at the following URL:

  `http://www.ibm.com/servers/eserver/zseries/zos/suf/`

We recommend that you review the ShopzSeries option, once it becomes available in your geography, because most of the capabilities of the other maintenance offerings for preventive and corrective, both physical and electronic, are available through ShopzSeries. While end of support for SUF has not been announced, we expect that for z/OS platform service, SUF will be functionally stabilized and future enhancements will be made to ShopzSeries.

22.4.5 Enhanced HOLDDATA

IBM provides Enhanced HOLDDATA, which is HOLDDATA with additional information to identify the reason for the hold, and a fixing PTF. Enhanced HOLDDATA provides a hold against the FMID for HIPER maintenance. For PEs, it provides a hold against the PTF in error. Enhanced HOLDDATA can help manage all products on the OS/390 or z/OS platform.

Enhanced HOLDDATA is received into the SMP/E Global zone. You can use the SMP/E REPORT ERRSYSMODS command on any Target zone to identify missing critical service that applies to your system. This allows you to identify any missing HIPER or PE fixes. Additionally, the report identifies whether a corrective PTF is available, whether the corrective PTF is already in RECEIVE status, and any symptom flags for a HIPER.

Enhanced HOLDDATA is available through ESO packages, with CBPDO, and from the Web. For more information on Enhanced HOLDDATA, visit the Web site:

`http://service.boulder.ibm.com/390holddata.html`

If you have chosen to implement a GoldPlex or PlatinumPlex, we recommend merging into a single Global zone so the Enhanced HOLDDATA would only need to be received once. While this would also be desirable in a BronzePlex, limited DASD sharing and catalog issues may make this impossible to achieve.

New HOLD SYSTEM Reason IDs

In addition to the existing HOLD SYSTEM reason IDs (ACTION, AO, DELETE, DEP, DOC, EC, EXRF, IOGEN, and MSGSKEL), several new HOLD SYSTEM reason IDs have been added (DB2BIND, ENH, EXIT, IPL, RESTART, DDDEF, DOWNLD, and MULTSYS). These new reason IDs will be used where only ACTION HOLDs were used previously.

These new REASON IDs allow you more granularity in the ability to bypass SYSTEM HOLDs when you know you will perform the documented action, or when you know the action does not apply to your environment. For instance, if you know you plan to perform an IPL with a CLPA after service is installed, a BYPASS HOLDSYS(IPL) could be coded during the install. There would be no need to read HOLDDATA for SYSTEM HOLDs with a Reason ID of IPL.

Conversely, you may choose to wait and install service containing a particular HOLD reason until you are ready to take the specified action or make the indicated change on your system. With the new HOLD reason of ENH (enhancement), the installation of new function could be done at a time other than the installation of regular preventive service. By not coding a BYPASS HOLDSYS(ENH), any new-function PTFs would not be installed.

There are no SMP/E changes required to recognize these new REASON IDs. Any release of SMP/E will work. Implementation of these new REASON IDs by IBM will roll out as internal processes are updated. There is no defined order for the products or elements to implement these new REASON IDs. The new HOLD SYSTEM REASON IDs and their definitions are as follows:

- **DB2BIND**

  A DB2 application REBIND is required to activate the change incorporated in the PTF.

- **ENH**

  The PTF contains an enhancement to the existing product. This is normally used with new-function PTFs.

- **EXIT**

  The PTF contains a change to a sample user exit or interface.

- **IPL**

  The PTF requires a system IPL with special requirements (for example, IPL with CLPA) after the application of the PTF in order for the PTF to become effective.

- **RESTART**

  This PTF requires special instructions regarding subsystem restart. This does not include service that requires normal subsystem restart to be effective.

- **DDDEF**

  The PTF requires a change to one or more DDDEF definitions.

- **DOWNLD**

  The PTF delivers code that needs to be downloaded.

- **MULTSYS**

  Used to identify PTFs that have special installation requirements in a multisystem environment. These include:

  - Preconditioning - to identify maintenance that requires other maintenance to be installed on other systems in a complex before this maintenance can be installed or deployed.
  - Coexistence (toleration) - to identify maintenance that requires other maintenance to be installed on other systems in a complex before new function provided in this PTF can be installed or deployed.

    In exception conditions, a PTF may be considered a coexistence PTF if it is used to identify other maintenance to be installed on other systems in a complex, before function originally shipped in the product (FMID) can be deployed. This would be limited to cases where the need for coexistence service wasn't known when the product (FMID) was originally made available.
  - Complete fix (exploitation) - to identify maintenance that needs to be installed and deployed on multiple systems before the change can be effective.

22.5 The maintenance environment

There are many ways you could set up a maintenance environment and propagate service between systems within the target sysplex. We will provide suggested ways of setting up the maintenance environment. These suggestions could be used, either in part or in full, to maintain your maintenance environment for systems within the target sysplex.

The suggested structure would be best suited for a PlatinumPlex or a GoldPlex, but could be set up for a BronzePlex configuration as well. The suggested structure outlines an environment for the operating system-related products, but could potentially be used for any program product or subsystem.

22.5.1 Suggested structure

We recommend keeping the maintenance environment structure as simple as possible, within the bounds of giving you the flexibility you require. You should develop and document the maintenance structure, maintenance and upgrade policies, and naming standards. Keeping it as simple as possible will help ensure compliance to the process.

You will need to set up your maintenance environment to reflect the target and distribution structure as per the sysres structure. We recommend that all target libraries reside on the sysres volumes and all distribution libraries reside on other non-sysres volumes, preferably SMS-managed. We also recommend that, in a GoldPlex or BronzePlex configuration, you maintain your maintenance environment from a single system within the target sysplex to avoid any user catalog or SMSplex issues.

Figure 22-2 on page 380 outlines the structure of the target sysplex that we will be using as an example in this section. There are four systems sharing a sysres structure. There are three primary sysres volumes (PLXSY1, PLXSY2, and PLXSY3) that each of the four systems could be IPLed off. A single maintenance environment is used, with a propagation routine to build the sysres at the required level, including a snapshot of the appropriate SMP/E data sets to reflect the sysres level. The SMP/E data sets used by the maintenance environment are SMS-managed.

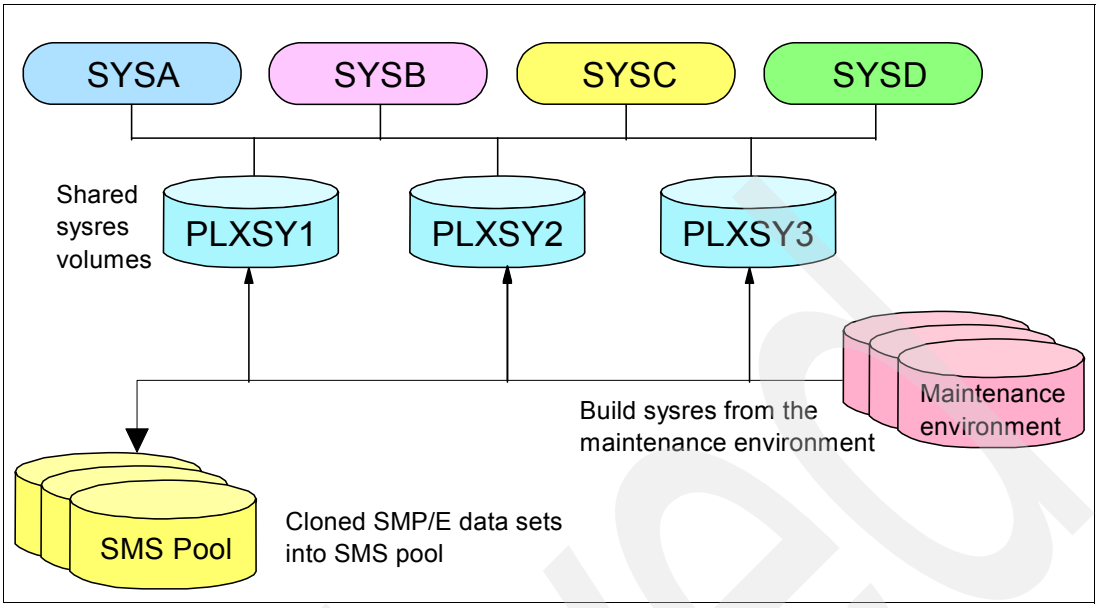


Figure 22-2 Suggested maintenance structure

Naming standards

Naming standards within the maintenance structure are very important. They need to be simple so that people will adhere to them, but flexible enough to cater for differing product levels. The list of naming standards in Table 22-3 is focused on z/OS and related program products, but could also be used in other areas.

The GoldPlex and BronzePlex configurations do not include the sharing of the Master Catalog within the target sysplex, and therefore user catalogs may or may not be shared. So you will need to consider how you want to manage the maintenance environment's data set catalog entries.

The options would be to either maintain your maintenance environment from a single system within the target sysplex, or make the HLQ(s) for the data sets within the maintenance environment available in a user catalog that is shared by all systems within the sysplex. If you choose the latter option, there may be issues if your maintenance environments data sets are SMS-managed.

In a PlatinumPlex configuration, there would be a single Master Catalog with a single SMSplex, so managing the data sets with SMS would not cause an issue.

Table 22-3 Example naming standards for the maintenance structure

Type	Standard	Example
Product Group	xxxvrm where xxx is a 3-character product identifier and vrm is the version, release modification of the product.	OS#2A0 for OS/390 2.10 ZOS130 for z/OS 1.3
Global Zone	SMPE.GLOBAL.CSI	SMPE.GLOBAL.CSI

Type	Standard	Example
Target Zones	<xxxvrm>T for the maintenance zone, one per product group. <resvol>T one per sysres.	ZOS130T PLXSY1T
Distribution Zone	<xxxvrm>D one per product group.	ZOS130D
Target CSIs	SMPE.<xxxvrm>.TLIB.CSI for the maintenance zone. SMPE.<xxxvrm>.<resvol>T.CSI for the sysres zones.	SMPE.ZOS130.TLIB.CSI SMPE.ZOS130.PLXSY1T.CSI
Distribution CSI	SMPE.<xxxvrm>.DLIB.CSI	SMPE.ZOS130.DLIB.CSI
SMP/E data sets	SMPE.GLOBAL.SMPPTS a single SMPPTS SMPE.GLOBAL.SMPLOG* SMPE.<xxxvrm>.TLIB.SMP* for the maintenance zone. SMPE.<xxxvrm>.<resvol>T.SMP* for the sysres zones.	 SMPE.GLOBAL.SMPLOG/SMPLOGA SMPE.ZOS130.TLIB.SMPLTS SMPE.ZOS130.PLXSY1T.SMPMTS
Target data sets	SYS1.<dddef>	SYS1.LINKLIB, with the DDDEF pointing to the appropriate target zone volume.
Distribution data sets	SMPE.<xxxvrm>.<dddef>	SMPE.ZOS130.ALINKLIB

We have used SMPE as the HLQ in the examples, but this could obviously be changed to suit your site's requirements.

SMP/E structure

The naming standards listed in Table 22-3 allow for a single global zone, one target zone each for the maintenance libraries and for each sysres set (to ensure there is a maintenance environment that reflects the sysres level), and a single distribution zone.

This structure allows for future releases and, with slight naming standard changes, could also cater for different maintenance strategies. For example, when a new level of z/OS is installed onto the system, a new SMP/E infrastructure would need to be introduced. When the new level is rolled through all the systems within the target sysplex, the old structure could then be removed.

Table 22-4 shows an example of the SMP/E structure based on the naming standards described. The structure shows the single global, maintenance environment target and distribution zones, and target zones, for each sysres set. These target zones get refreshed by the propagation routine and would be used for level information specific to the sysres, but could also be used to apply maintenance directly, if required.

Table 22-4 Example SMP/E structure

Zone type	Zone name	CSI data set name
Global	GLOBAL	SMPE.GLOBAL.CSI
Target	ZOS130T	SMPE.ZOS130.TLIB.CSI

Zone type	Zone name	CSI data set name
Target	PLXSY1T	SMPE.ZOS130.PLXSY1.CSI
Target	PLXSY2T	SMPE.ZOS130.PLXSY2.CSI
Target	PLXSY3T	SMPE.ZOS130.PLXSY3.CSI
Distribution	ZOS130D	SMPE.ZOS130.DLIB.CSI

Another alternative would be to remove the maintenance environment target zone (ZOS130T) and simply apply your maintenance directly to the sysres target zones. This would be a valid configuration as long as the maintenance was applied to an inactive sysres.

Note: *Never* apply maintenance to a sysres that is active within the target sysplex. We discuss how to prevent this from happening in 22.5.4, “Propagation of service and releases within the sysplex” on page 385.

DDDEFs

The DDDEFs within the target and distribution zones will need to be set up to reflect the maintenance structure. The target library DDDEF entries need to have the specific target volume specified, as shown in Example 22-1.

Example 22-1 Target library DDDEF example

```
Entry Type:  DDDEF                                Zone Name: PLXSY3T
Entry Name:  LINKLIB                              Zone Type: TARGET

DSNAME: SYS1.LINKLIB

VOLUME: PLXSY3        UNIT: SYSALLDA  DISP:   SHR
```

For HFS data sets, the target path DDDEF entries need to have the service and zone directory structure specified, as in Example 22-2. Automount can be used to mount the correct root HFS with the appropriate target zone name; refer to **22.5.2, “HFS considerations” on page 383** for more information on the automount set up.

Example 22-2 Target path DDDEF example

```
Entry Type:  DDDEF                              Zone Name: Z0S130T
Entry Name:  NFSCUTIL                           Zone Type: TARGET

-------------------------------------------------------------------
PATH: '/SERVICE/Z0S130T/usr/lpp/NFS/IBM/'
```

The distribution library DDDEF entry needs to have the cataloged data set name specified as shown in Example 22-3. We recommend these data sets be SMS-managed.

Example 22-3 Distribution library DDDEF example

```
Entry Type:  DDDEF                               Zone Name: Z0S130D
Entry Name:  ALINKLIB                            Zone Type: DLIB

DSNAME: SMPE.ZOS130.ALINKLIB

VOLUME:              UNIT: SYSALLDA  DISP:   SHR
```

22.5.2 HFS considerations

With each release of OS/390 and z/OS, IBM supplies a root HFS data set containing files and executables for the operating system elements. To avoid confusion with the sysplex root HFS data set, which is discussed in Chapter 8, “Shared HFS considerations” on page 115, we will call the IBM-supplied root HFS the “Version HFS”.

A copy of the Version HFS is required for the maintenance environment, as well as for each of the sysres sets. The placement of the Version HFS is flexible—it could be on or off the sysres, and SMS- or non-SMS-managed. We recommend placing the Version HFS on one of the sysres overflow volumes for ease of management, especially in a GoldPlex configuration, because of the limited DASD sharing and separate SMSplexes.

Naming standards for the HFS data sets

Naming standards within the maintenance structure are very important. They need to be simple to define and identify, but flexible enough to cater for differing product levels. The naming standards in Table 22-5 are focused on the Version HFS.

Table 22-5 Example naming standards for the Version HFS

Mountpoint	Data set name	Example	Comments
/$VERSION	OMVS.<resvol>.ROOT	OMVS.PLXSY1.ROOT	Mounted read-only in BPXPRMxx
/SERVICE/tzone	OMVS.<tzone>.ROOT OMVS.<resvol>.ROOT	OMVS.ZOS130T.ROOT OMVS.PLXSY2.ROOT	Mounted read/write - automounted when required for maintenance.

Automount for service

The main reason for using automount for the /SERVICE directory is so that the required Version HFS data sets are mounted only when required, and SMP/E maintenance can be applied to either the maintenance target zone or any of the sysres volumes, if required. This will allow the root HFS to be mounted on the system where the SMP/E job is being run, as long as it isn't already mounted on another system. SMP/E will mount the HFS associated with the DDDEF specified in the SMP/E zone, as in Example 22-2, “Target path DDDEF example” on page 382.

Automount can manage the /SERVICE directory by setting up the auto.master file in the /etc directory; see Example 22-4.

Example 22-4 /etc/auto.master example

```
/SERVICE            /etc/SERVICE.map
```

The corresponding map needs to be set up to tell automount which HFS to mount on the appropriate mountpoint; see Example 22-5.

Example 22-5 /etc/SERVICE.map example

```
name                ZOS130T
type                HFS
filesystem          OMVS.ZOS130T.ROOT
mode                RDWR
duration            60
delay               0
setuid              no
```

```
name                 PLXSY1T
type                 HFS
filesystem           OMVS.PLXSY1.ROOT
mode                 RDWR
duration             60
delay                0
setuid               no

name                 PLXSY2T
type                 HFS
filesystem           OMVS.PLXSY2.ROOT
mode                 RDWR
duration             60
delay                0
setuid               no

name                 PLXSY3T
type                 HFS
filesystem           OMVS.PLXSY3.ROOT
mode                 RDWR
duration             60
delay                0
setuid               no
```

The automount filesystype entry also needs to be set up in BPXPRMxx, if it isn't already, as in Example 22-6.

Example 22-6 Automount filesystype entry in BPXPRMxx

```
FILESYSTYPE TYPE(AUTOMNT)
            ENTRYPOINT(BPXTAMD)
```

To ensure the automount facility is started after each IPL, the /etc/rc file on the appropriate system will need to be updated as per Example 22-7.

Example 22-7 Start automount in /etc/rc

```
# Start the automount facility
echo "starting AUTOMOUNT" > /dev/console
/usr/sbin/automount
```

With the automount structure set up as in the examples, whenever maintenance is applied to any of the target zones, the appropriate Version HFS will be mounted. For example, if maintenance is being applied to the ZOS130T zone and the process was modifying a file in a path specified in a DDDEF starting with /SERVICE/ZOS130T/*, then automount would automatically mount the Version HFS OMVS.ZOS130T.ROOT at the /SERVICE/ZOS130T mountpoint. As per the parameters specified in Example 22-5 on page 383, it would be mounted Read/Write for a duration of 60 minutes.

22.5.3 Cross-product and cross-system requisite checking

Some of the most labor-intensive and error-prone tasks involved in deploying a new level of a product (or system) are the identification, verification, and possibly, installation of requisite coexistence (sometimes called *preconditioning*) service. The same can be said for cross-product dependencies, especially for those products maintained in separate SMP/E zones, possibly with separate global zones.

Setting up automatic cross-zone requisite checking can reduce or eliminate these tasks. This process could, for example, automatically verify that all coexistence service is installed when installing a new level of the operating system. Or, the process can ensure all cross-product dependencies (for example, OS/390 or z/OS PTFs, DB2 PTFs, and so on) are installed when installing products like WebSphere 4.0.1.

Cross-zone checking

In OS/390 Release 3, SMP/E introduced the capability to automate the checking of cross-zone requisites. These cross-zone requisites can be for cross-product dependencies on the same system, as well as for cross-system “Preconditioning”, “Coexistence” (“toleration”), or “Completing a fix” (“exploitation”) PTFs. Product packagers can use ++IFREQs SMP MCS statements to identify these requisites.

Different methods can be used for cross-zone processing. However, you will need to set up your SMP/E environment appropriately. A zone group can be defined and added to the install jobs, or the XZGROUP operand can be used.

Once set up, SMP/E can identify cross-zone requisites needed in the “set-to zone” which is in effect for the current APPLY/ACCEPT commands, as well as any cross-zone requisites for SYSMODs currently being installed. SMP/E checks if the requisite is already installed, or if it needs to be installed as part of the same SMP/E APPLY/ACCEPT command. Once products use ++IFREQs for MULTSYS PTFs, SMP/E will be able to verify and install cross-zone requisites, thereby satisfying the ++HOLD REASON(MULTSYS) exception condition.

Note: If SYSMODs being installed into the set-to zone have requirements against the other cross-zones, that service must be APPLYed to those zones before installation can be completed into the set-to zone.

For detailed information on how to set up automatic cross-zone requisite checking, refer to “Specifying Automatic Cross-Zone Requisite® Checking” in Chapter 3, “Preparing to Use SMP/E” in *SMP/E for z/OS and OS/390 User’s Guide,* SA22-7773.

22.5.4 Propagation of service and releases within the sysplex

The ability to quickly and easily clone software libraries and the associated SMP/E information is key to reducing the manual overhead involved in software management. You will need to set up a cloning or propagation routine to do this, if you haven’t already. In this section, we provide a list of the steps required for this task, and examples of each step.

The steps and examples given are based on building a sysres (for example, PLXSY1) from the maintenance environment (ZOS130T, for example) described in 22.5.1, “Suggested structure” on page 379. We included a snapshot of the SMP/E environment.

Check for active sysres

In a shared sysres environment, it is important to ensure the cloning or propagation process does not try to overwrite a sysres that is active within the target sysplex. Obviously, checking the target sysres prior to running the propagation routine is required, but mistakes can happen. One way of automating this is presented in Example 22-8 on page 386.

The REXX program issues: RO *ALL,D U,VOL=target_sysres, and then traps the output and checks if the target_sysres is active on any systems within the target sysplex. If it is, the program exits with a RC=20. This program, or something similar, can be executed as the first step of the propagation routine to help safeguard against human error. This program assumes you have access to the CONSOLE command in TSO.

Example 22-8 Sample program to check for active sysres

```
/*REXX ---------------------------------------------------------------*/
/*                                                                   */
/* SYNTAX:                                                           */
/*                                                                   */
/*  TSO CHKRES   {<wait>{!}} <sysres>                                */
/*                                                                   */
/*    <wait>    - Number of seconds to wait for a response         */
/*    !         - Waits the entire wait time to trap multiple WTOs   */
/*                in response to command. If not used then the first */
/*                WTO is trapped and the console sess'n is completed */
/*    <sysres>  - MVS command to be issued:                          */
/*                RO *ALL,D U,VOL=&sysres                            */
/*                                                                   */
/* OUTPUT:                                                           */
/*                                                                   */
/*    Series of lines written to the terminal.                       */
/*                                                                   */
/*-------------------------------------------------------------------*/
arg wait sysres
trace o
wait=strip(wait)
sysres=strip(sysres)
w1=reverse(substr(reverse(wait),2))
w2=right(wait,1)
if datatype(w1,'w')&w2="!" then
 do
  wait=w1
  waititout=1
 end
else
 if datatype(wait,"W") then
  waititout=0
 else
  do
   sysres=wait sysres
   wait=5
   waititout=0
  end
soldisp=sysvar("SOLDISP")
unsdisp=sysvar("UNSDISP")
solnum=sysvar("SOLNUM")
unsnum=sysvar("UNSNUM")
"consprof soldisplay(no) unsoldisplay(no) solnum(5000) unsolnum(5000)"
if rc¬=0 then exit 20
cart="C"time('s')
"console activate name("cart")"
if rc=0 then
 do
  "console syscmd(RO *ALL,D U,VOL="sysres") cart("cart")"
  if waititout then
   do until mmm¬=0
    out.0=0
    mmm=getmsg("out.","SOL",cart,,wait)
    if mmm=0 then
     do i=1 to out.0
      parse var out.i word1 word2 word3 word4 word5
      status=" "
      if word4=sysres then
       do
```

```
        reschk=substr(word3,1,1)
        if reschk="S" then ok="NOTOK"
        if ok="NOTOK" then status="NOTOK"
       end
      say out.i status
     end
   end
  else
   do
    out.0=0
    mmm=getmsg("out.","SOL",cart,,wait)
    if mmm=0 then
     do i=1 to out.0
      parse var out.i word1 word2 word3 word4 word5
      status=" "
      if word4=sysres then
       do
        reschk=substr(word3,1,1)
        if reschk="S" then ok="NOTOK"
        if ok="NOTOK" then status="NOTOK"
       end
      say out.i status
     end
   end
  "console deactivate"
 end
else
 say "CONSOLE NOT AVAILABLE"
"consprof soldisplay("soldisp")",
         "unsoldisplay("unsdisp")",
         "solnum("solnum")",
         "unsolnum("unsnum")"
if ok="NOTOK" then exit 20
exit 0
```

Propagation routine

These steps and examples are based on building a sysres from the maintenance environment, including creating a snapshot of the SMP/E environment. The following assumptions are made:

- A 3390 model 9 device is used for all sysreses, meaning that all sysres data sets fit on a single volume. If your “sysres” actually consists of more than one volume (for example, if you are using 3390 model 3 devices), you will need to adjust the sample jobs accordingly.
- All the sysres data sets for the maintenance environment are actually kept on a single device. While you could spread these data sets over a number of devices, mixed with other data sets if you wish, there are benefits to keeping all the data sets together:
  - It is easier to create the production IPL volumes if all the source data sets are grouped together (you can use a full-pack copy).
  - If you need to test an emergency fix, the maintenance environment data sets can actually be IPLed in a test LPAR.

The steps required are:

1. Verify that the target sysres is not active on any other systems within the sysplex.

   This step uses the REXX exec in Example 22-8 on page 386 to check if the target_sysres is active on any systems within the target sysplex. If it is, the program exits with a RC=20. As long as the condition code checking on the following steps are coded accordingly, all following steps should flush and the job ends.

Example 22-9 Propagation - verify target sysres is not active

```
//CHKRES   EXEC PGM=IKJEFT01,REGION=0M
//SYSEXEC  DD DISP=SHR,DSN=SYS1.plex_name.EXEC
//SYSPRINT DD SYSOUT=Z
//SYSABEND DD SYSOUT=Z
//SYSTSPRT DD SYSOUT=*
//SYSIN    DD DUMMY
//SYSTSIN  DD *
  CHKRES 30 target_sysres
/*
```

2. Initialize the target sysres while online.

   This step is needed to clear the target sysres volume(s) in preparation for copying the required sysres data sets. It will initialize the volume(s) without having to vary the device(s) offline, therefore saving manual intervention during the propagation process. If your sysres consists of multiple volumes, you should initialize all volumes.

Example 22-10 Propagation - Initialize the target sysres

```
//INITRES  EXEC PGM=ICKDSF,COND=(0,LT)
//SYSPRINT DD SYSOUT=*
//RESVOL   DD DISP=SHR,UNIT=3390,VOL=SER=target_sysres
//SYSIN    DD *
  INIT DDNAME(RESVOL) -
  DEVICETYPE(3390) -
  VFY(target_sysres) -
  INDEX(0,1,29) -
  VOLID(target_sysres) -
  PURGE -
  NOCHECK -
  VTOC(2,0,150)
/*
```

3. Copy the non-OMVS sysres data sets.

   This step, shown in Example 22-11, copies all the required target data sets from the maintenance sysres to the target sysres, *excluding* the OMVS Version root HFS.

Example 22-11 Propagation - copy sysres data sets

```
//COPYRES  EXEC PGM=ADRDSSU,REGION=0M,COND=(4,LT)
//INDD     DD UNIT=SYSALLDA,DISP=SHR,VOL=SER=maint_sysres
//OUTDD    DD UNIT=SYSALLDA,DISP=SHR,VOL=SER=target_sysres
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
  COPY DATASET(INCLUDE(                          -
                       **                        -
                       )                         -
               EXCLUDE(                          -
                       OMVS.maint_zone.ROOT -
                       SYS1.VTOCIX.**            -
                       SYS1.VVDS.**              -
```

```
                        )                                  -
                )                                          -
        INDD(INDD)                                         -
        OUTDD(OUTDD)                                       -
        PROCESS(SYS1)                                      -
        ALLEXCP                                            -
        TOL(ENQF)                                          -
        ADMIN
/*
```

4. Delete the target Version HFS catalog entry.

   Prepare for the replacement of the target Version HFS. This data set needs to be directly cataloged on the sysres (or one of the overflow volumes, if you have a multi-volume sysres) to ensure automount can find the HFS on the correct volume when required.

Example 22-12 Propagation - delete target Version root HFS catalog entry

```
//DELROOT  EXEC PGM=IDCAMS,COND=(4,LT)
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
  DEL 'OMVS.target_sysres.ROOT' NOSCRATCH
  SET MAXCC=0
/*
```

5. Copy the OMVS Version HFS data set.

   This step uses a DFSMSdss dump/restore process to copy, rename, and catalog the HFS.

Tip: If you are using z/OS 1.3 or later, it is possible to do a logical data set copy of the HFS data set, rather than having to do a DUMP/RESTORE as shown here.

Example 22-13 Propagation - copy Version root HFS data set

```
//DMPOMVS  EXEC PGM=ADRDSSU,REGION=0M,COND=(4,LT)
//SYSPRINT DD SYSOUT=*
//TMPDASD  DD DSN=&&DMPOMVS,DISP=(NEW,DELETE),
//            UNIT=SYSDA,SPACE=(CYL,(500,500))
//OUTDASD  DD DISP=SHR,UNIT=3390,VOL=SER=target_sysres
//SYSIN    DD *
  DUMP    DATASET(INCLUDE(                               -
                        OMVS.maint_zone.ROOT     -
                                             ))          -
          TOL(ENQF)                                      -
          OUTDDNAME(TMPDASD)
  RESTORE INDDNAME(TMPDASD)                              -
          DATASET(INCLUDE(**))                           -
          RENUNC((OMVS.maint_zone.ROOT,OMVS.target_sysres.ROOT)) -
          OUTDD(OUTDASD)                                 -
          CATALOG                                        -
          TOL(ENQF)                                      -
          SHR
/*
```

6. Create IPL text at the latest release level on the target sysres.

Example 22-14 Propagation - create IPL text on the target sysres

```
//IPLTEXT  EXEC PGM=ICKDSF,COND=(4,LT)
//SYSPRINT DD SYSOUT=*
//IPLTEXT  DD DISP=SHR,DSN=SYS1.SAMPLIB(IPLRECS),
```

```
//            UNIT=3390,VOL=SER=target_sysres
//         DD DISP=SHR,DSN=SYS1.SAMPLIB(IEAIPL00),
//            UNIT=3390,VOL=SER=target_sysres
//RESVOL   DD DISP=SHR,UNIT=3390,VOL=SER=target_sysres
//SYSIN    DD *
  REFORMAT DDNAME(RESVOL) -
  DEVICETYPE(3390) -
  VFY(target_sysres) -
  VOLID(target_sysres) -
  IPLDD(IPLTEXT,OBJ) -
  BOOTSTRAP
/*
```

7. Create the Master Catalog entry in nucleus. You would normally use the SYSCAT parameter in the LOADxx member to specify the Master Catalog name. (However, if for some reason you still use an “A” in your Loadparm and the operator presses <Enter> at the prompt `Specify master catalog parameter` during the IPL, the SYSCATLG member of Nucleus will be used to ensure the IPL does not fail.)

Example 22-15 Propagation - create Master Catalog entry

```
//SYSCATLG EXEC PGM=IEBGENER,COND=(4,LT)
//SYSPRINT DD SYSOUT=*
//SYSIN    DD DUMMY
//SYSUT2   DD DISP=SHR,DSN=SYS1.NUCLEUS(SYSCATLG),
//            UNIT=3390,VOL=SER=target_sysres
//SYSUT1   DD *
volser133Cmaster_catalog_name
/*
```

8. Delete SMP/E data sets related to the previous level of the target sysres.

   This step deletes the old SMP/E data sets related to the target sysres volume’s target zone, in preparation for a copy. An example of such a data set is: SMPE.ZOS130.PLXSY1T.SMPLTS.

Example 22-16 Propagation - delete target SMP/E data sets

```
//*--------------------------------------------------------------------
//*        Delete SMPE data sets
//*--------------------------------------------------------------------
//DELSMPE  EXEC PGM=IDCAMS,COND=(4,LT)
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
  DEL 'SMPE.xxxvrm.target_sysresT.SMPLTS'
  DEL 'SMPE.xxxvrm.target_sysresT.SMPMTS'
  DEL 'SMPE.xxxvrm.target_sysresT.SMPSCDS'
  DEL 'SMPE.xxxvrm.target_sysresT.SMPSTS'
  DEL 'SMPE.xxxvrm.target_sysresT.SMPLOG'
  DEL 'SMPE.xxxvrm.target_sysresT.SMPLOGA'
  SET MAXCC=0
/*
```

9. Create the SMP/E target data sets related to the target sysres. These data sets would normally be SMS-managed and would *not* be placed on the sysres.

   This step takes a snapshot of the maintenance zone’s SMP/E data sets and copies them to the SMP/E data sets related to the target sysres.

Note: The sample jobs assumes that the output data sets will be SMS-managed. If this is not the case, you must add a DD statement referring to the target device, and an OUTDD() DSS control statement referring to that DD statement.

Example 22-17 Propagation - copy target SMP/E data sets

```
//COPYSMPE EXEC PGM=ADRDSSU,REGION=0M,COND=(4,LT)
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
  COPY DATASET(INCLUDE(                                -
                       SMPE.xxxvrm.TLIB.SMPLTS         -
                       SMPE.xxxvrm.TLIB.SMPMTS         -
                       SMPE.xxxvrm.TLIB.SMPSCDS        -
                       SMPE.xxxvrm.TLIB.SMPSTS         -
                       SMPE.xxxvrm.TLIB.SMPLOG         -
                       SMPE.xxxvrm.TLIB.SMPLOGA        -
                       )                               -
               )                                       -
       RENUNC((SMPE.xxxvrm.TLIB.**,SMPE.xxxvrm.target_sysresT.**)) -
       CATALOG                                         -
       REPLACE                                         -
       TOL(ENQF)                                       -
       ADMIN
/*
```

10. Delete the target CSI for the target sysres zone.

    This step deletes the target CSI related to the target sysres in preparation for the rebuild of the CSI.

Example 22-18 Propagation - delete target sysres CSI

```
//DELCSI   EXEC PGM=IDCAMS,COND=(4,LT)
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
  DEL 'SMPE.xxxvrm.target_sysresT.CSI' CLUSTER PURGE
  SET MAXCC=0
/*
```

11. Define the target CSI for the target sysres zone.

    This step defines the target zone CSI that relates to the target sysres in preparation for the zonecopy.

    Note: The sample jobs assume that the output data sets will be SMS-managed. If this is not the case, you must add a VOLUME parameter referring to the target device.

Example 22-19 Propagation - define target sysres CSI

```
//DEFCSI   EXEC PGM=IDCAMS,COND=(4,LT)
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
  DEFINE CLUSTER (NAME(SMPE.xxxvrm.target_sysresT.CSI) -
                  FREESPACE(10 5)             -
                  KEYS(24 0)                  -
                  RECORDSIZE(24 143)          -
                  SHAREOPTIONS(2 3)           -
                  UNIQUE)                     -
          DATA    (NAME(SMPE.xxxvrm.target_sysresT.CSI.DATA) -
                  CISZ(4096)                      -
                  CYLINDERS(150 20))              -
          INDEX   (NAME(SMPE.xxxvrm.target_sysresT.CSI.INDEX) -
                  TRACKS(5 5))
/*
```

12. Initialize the target zone CSI for the target sysres zone.

Example 22-20 Propagation - initialize the target zone CSI

```
//INITCSI  EXEC PGM=IDCAMS,COND=(4,LT)
//SYSPRINT DD SYSOUT=*
//TZONE    DD DSN=SMPE.xxxvrm.target_sysresT.CSI,DISP=OLD
//ZPOOL    DD DSN=SYS1.MACLIB(GIMZPOOL),DISP=SHR
//SYSIN    DD *
  REPRO OUTFILE(TZONE) INFILE(ZPOOL)
/*
```

13. Copy the zone from the maintenance target zone to the target sysres zone.

    This step uses the SMP/E zonecopy command to copy the maintenance target zone into the target zone related to the sysres.

Example 22-21 Propagation - Copy target zone

```
//ZONECOPY EXEC SMPE,COND=(4,LT)
//SMPCSI   DD DISP=SHR,DSN=SMPE.GLOBAL.CSI
//SMPLOG   DD DISP=SHR,DSN=SMPE.GLOBAL.SMPLOG
//SMPLOGA  DD DISP=SHR,DSN=SMPE.GLOBAL.SMPLOGA
//SYSIN    DD *
  SET
    BOUNDARY(target_sysresT) .
    ZONECOPY (xxxvrmT) INTO(target_sysresT) .
/*
```

14. Zone edit the target sysres zone.

    This step uses the SMP/E ZONEEDIT command to change all references of the volume and path entries from the maintenance zone to the target sysres.

Example 22-22 Propagation - edit target zone

```
//ZONEEDIT EXEC SMPE,COND=(4,LT)
//SMPCSI   DD DISP=SHR,DSN=SMPE.GLOBAL.CSI
//SMPLOG   DD DISP=SHR,DSN=SMPE.GLOBAL.SMPLOG
//SMPLOGA  DD DISP=SHR,DSN=SMPE.GLOBAL.SMPLOGA
//SYSIN    DD *
  SET
    BOUNDARY(target_sysresT) .
    ZONEEDIT DDDEF.
     CHANGE VOLUME(maint_sysres,target_sysres).
     CHANGE PATH('/SERVICE/xxxvrmT/'*,'/SERVICE/target_sysresT/'*).
    ENDZONEEDIT.
/*
```

22.6 Methodology for merging maintenance environments

When moving an incoming system into the target sysplex, the methodology for merging the maintenance environment will depend on the features you plan to exploit within the sysplex.

22.6.1 Merge global zones

You can use the SMP/E GZONEMERGE command to copy information from one SMP/E global zone to another. This allows you to reduce the number of global zones that you must manage within the target sysplex.

Before merging one global zone into another, we recommend that you first clean up the originating global zone and its related SMPPTS, in order to reduce the amount of data to be merged. Do this by deleting all unneeded SYSMOD and HOLDDATA entries from the global zone, and deleting unneeded MCS entries from the SMPPTS. You can use the REJECT NOFMID and REJECT PURGE SMP/E commands to delete those entries related to PTFs and FMIDs that are no longer needed.

Also, rather than manually determining which FMIDs to delete, you can use the sample programs GIMCRSAM and GIMPRSAM (provided in SYS1.SAMPLIB) to help you create a REJECT NOFMID command for the FMIDs that are superseded or deleted in the DLIB zones you specify. Refer to *SMP/E for z/OS and OS/390 Commands,* SA22-7771, for more information on the GZONEMERGE command.

You can use the GZOMEMERGE command to copy information for selected FMIDs or for the entire contents of a global zone. For example, to merge the entire contents from one global zone to another, you would use the statements contained in Example 22-23, “SMP/E Statements for GZONEMERGE” on page 393.

Example 22-23 SMP/E Statements for GZONEMERGE

```
SET BDY(GLOBAL)                          /* Set to the destination Global zone   */
.
GZONEMERGE
FROMCSI(from.global.zone.data.set.CSI) /* From Global zone CSI                    */
CONTENT                                /* Indicates SYSMODs and corresponding
                                          SMPPTS members                         */
DEFINITION                             /* Indicates OPTIONS, UTILITY, ZONESET,
                                          FMIDSET, GZONE Entry stuff, etc. are
                                          to be copied                           */
.
```

22.7 Tools and documentation

In this section we provide information about tools and documentation that may help you complete this task.

22.7.1 Tools

GZONEMERGE

You can use the SMP/E GZONEMERGE command to copy information from one SMP/E global zone to another. This allows you to reduce the number of global zones that you must manage within the target sysplex. See 22.6.1, “Merge global zones” on page 392 for more information.

22.7.2 Documentation

The following publications provide information that may be helpful in managing your maintenance environment:

- *Improvements in z/OS Service*, REDP0324 (draft RedPaper)
- *OS/390 Software Management Cookbook,* SG24-4775
- *OS/390 Planning for Installation,* GC28-1726
- *SMP/E for z/OS and OS/390 Commands,* SA22-7771
- *SMP/E for z/OS and OS/390 User’s Guide,* SA22-7773
- *z/OS Planning for Installation,* GA22-7504

A

Additional material

This redbook refers to additional material that can be downloaded from the Internet as described below.

Locating the Web material

The Web material associated with this redbook is available in softcopy on the Internet from the IBM Redbooks Web server. Point your Web browser to:

ftp://www.redbooks.ibm.com/redbooks/SG246818

Alternatively, you can go to the IBM Redbooks Web site at:

ibm.com/redbooks

Select the **Additional materials** and open the directory that corresponds with the redbook form number, SG246818.

Using the Web material

The additional Web material that accompanies this redbook includes the following files:

File name	*Description*
6818RACF.zip	Zipped Code Samples for RACF
6818OPC.zip	Zipped Code Samples for OPC

System requirements for downloading the Web material

The following system configuration is recommended:

Hard disk space:	5 MB minimum
Operating System:	Windows/UNIX
Processor:	Any
Memory:	64 MB

How to use the Web material

Create a subdirectory (folder) on your workstation, and unzip the contents of the Web material zip file into this folder.

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

IBM Redbooks

For information on ordering these publications, see “How to get IBM Redbooks” on page 399.

- *Controlling S/390 Processors Using the HMC,* SG24-4832
- *Converting to RMM - A Practical Guide,* SG24-4998
- *DFSMSrmm Primer,* SG24-5983
- *Enhanced Catalog Sharing and Management,* SG24-5594
- *Hierarchical File System Usage Guide,* SG24-5482
- *ICF Catalog Backup and Recovery: A Practical Guide,* SG24-5644
- *OS/390 MVS Multisystem Consoles Implementing Sysplex Operations,* SG24-4626
- *OS/390 Parallel Sysplex Configuration Volume 2: Cookbook,* SG24-5638
- *OS/390 Software Management Cookbook,* SG24-4775
- *OS/390 Version 2 Release 10 Implementation,* SG24-5976
- *Parallel Sysplex - Managing Software for Availability,* SG24-5451
- *Parallel Sysplex Operational Scenarios,* SG24-2079
- *S/390 Parallel Sysplex: Resource Sharing,* SG24-5666

Other resources

These publications are also relevant as further information sources:

- *CICS TS Intallation Guide,* GC34-5985
- *DCAF V1.3 Installation and Configuration Guide,* SH19-4068
- *DCAF V1.3 Users Guide,* SH19-4069
- *DFSMSdfp Storage Administration Reference*, SC26-7331
- *Hardware Management Console Guide,* GC38-0453
- *Hardware Management Console Operations Guide,* SC28-6809
- *IMS/ESA V6 Common Queue Server Guide and Reference,* SC26-9517
- *JES2 Multi-Access Spool in a Sysplex Environment,* GG66-3263
- *Hardware Configuration Definition (HCD) Scenarios,* SC33-7987
- *MVS System Management Facilities,* SA22-7630
- *OS/390 MVS Installation Exits,* SC28-1753
- *OS/390 Planning for Installation,* GC28-1726
- *System Automation for OS/390 Planning and Installation,* GC28-1549
- *Tivoli Workload Scheduler for z/OS Programming Interfaces,* SH19-4545

- *Tivoli Workload Scheduler Planning and Scheduling the Workload,* SH19-4546
- *z/OS DFSMS Access Methods Services for Catalogs,* SC26-7394
- *z/OS DFSMS:Managing Catalogs,* SC26-7409
- *z/OS DFSMS Migration,* GC26-7398
- *z/OS DFSMS Object Access Method Planning, Installation, and Storage Administration Guide for Tape Libraries,* SC35-0427
- *z/OS DFSMSdss Storage Administration Guide,* SC35-0423
- *z/OS DFSMShsm Diagnosis Reference,* LY35-0115
- *z/OS DFSMShsm Implementation and Customization Guide,* SC35-0418
- *z/OS DFSMShsm Storage Administration Reference*, SC35-0422
- *z/OS DFSMSrmm Implementation and Customization Guide,* SC26-7405
- *z/OS DFSMSrmm Reporting,* SC26-7406
- *z/OS Hardware Configuration Definition (HCD) Planning,* GA22-7525
- *z/OS Hardware Configuration Definition (HCD) Users Guide*, SC33-7988
- *z/OS ISPF Planning and Customizing,* GC34-4814
- *z/OS JES2 Installation Exits,* SA22-7534
- *z/OS JES2 Initialization and Tuning Guide,* SA22-7532
- *z/OS MVS Diagnosis: Tools and Service Aids,* GA22-7589
- *z/OS MVS Initialization and Tunning Reference,* SA22-7592
- *z/OS MVS Installation Exits,* SA22-7593
- *z/OS MVS Interactive Problem Control System (IPCS) User's Guide,* SA22-7596
- *z/OS MVS JCL Reference,* SA22-7597
- *z/OS MVS Planning: Global Resource Serialization,* SA22-7600
- *z/OS MVS Planning: Operations,* SA22-7601
- *z/OS MVS Programming: Assembler Services Guide,* SA22-7605
- *z/OS MVS System Commands,* SA22-7627
- *z/OS MVS System Messages Volume 9 (IGF-IWM),* SA22-7639
- *z/OS Planning for Installation,* GA22-7504
- *z/OS Security Server RACF Security Administrator's Guide,* SA22-7683
- *z/OS MVS Setting Up a Sysplex,* SA22-7625
- *zSeries 900 ESCON Channel-to-Channel Reference*, SB10-7034
- *zSeries 900 Processor Resource/Systems Manager Planning Guide,* SB10-7033

Referenced Web sites

These Web sites are also relevant as further information sources:

- CFSizer tool for calculating CF structure sizes:

  `http://www.ibm.com/servers/eserver/zseries/cfsizer/`
- List of CFLevel features and supported hardware:

  `http://www.ibm.com/servers/eserver/zseries/pso/cftable.html`

- Workload Manager home page:

  http://www.ibm.com/servers/eserver/zseries/zos/wlm/
- Washington Systems Center WLM migration checklist:

  http://www.ibm.com/servers/eserver/zseries/zos/wlm/pdf/wsc/pdf/v2.0_guide.pdf
- WLM Goal mode migration tool:

  http://www.ibm.com/servers/eserver/zseries/zos/wlm/migration/migration.html
- WLM Frequently Asked Questions

  http://www.ibm.com/servers/eserver/zseries/zos/wlm/faqs/faq00002.html#faqb9
- ThruPut Manager Web site:

  http://www.mvssol.com/
- Information about the use of the JES2 $D PERFDATA command:

  http://www.ibm.com/support/techdocs/atsmastr.nsf/PubAllNum/W9744B
- USS COPYTREE utility:

  http://www.ibm.com/servers/eserver/zseries/zos/unix/bpxa1ty2.html
- RACF tools Web site:

  http://www.ibm.com/servers/eserver/zseries/zos/racf/goodies.html
- RACF password copy tool:

  http://www.ibm.com/servers/eserver/zseries/zos/racf/pwdcopy.html
- RACF tool to synchronize RACF databases:

  http://www.ibm.com/servers/eserver/zseries/zos/racf/dbsync.html

How to get IBM Redbooks

You can order hardcopy Redbooks, as well as view, download, or search for Redbooks at the following Web site:

ibm.com/redbooks

You can also download additional materials (code samples or diskette/CD-ROM images) from that site.

IBM Redbooks collections

Redbooks are also available on CD-ROMs. Click the CD-ROMs button on the Redbooks Web site for information about all the CD-ROMs offered, as well as updates and formats.

Index

D

E

F

G

H

I

J

L

M

O

P

R

S

T

U

V

W

X

Z

Merging Systems into a Sysplex

(0.5” spine)
0.475”<->0.873”
250 <-> 459 pages

Merging Systems into a Sysplex

Redbooks

Position for PSLC and WLC benefits

Exploit System-level Resource Sharing

Maximize the benefits of sysplex

This IBM Redbook provides information to help Systems Programmers plan for merging systems into a sysplex. zSeries systems are highly flexibile systems capable of processing many workloads. As a result, there are many things to consider when merging independent systems into the more closely integrated environment of a sysplex. This book will help you identify these issues in advance and thereby ensure a successful project.

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

For more information:
ibm.com/redbooks

SG24-6818-00 ISBN 0738426083