IBM SMP/E for z/OS and OS/390

User's Guide

SA22-7773-04

IBM SMP/E for z/OS and OS/390

User's Guide

SA22-7773-04

Note!
Before using this information and the product it supports, be sure to read the general information under "Notices" on page 199.

Fifth Edition, August 2003

This book replaces the previous edition, SA22-7773-03, which is now obsolete. Changes or additions to text and illustrations are indicated by a vertical line to the left of the change.

This edition applies to IBM SMP/E for z/OS and OS/390, V3R2, program number 5655-G44, and to all subsequent releases and modifications, unless otherwise indicated in new editions.

Changes or additions to text and illustrations are indicated by a vertical line to the left of the change.

Order IBM publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address given below.

IBM welcomes your comments. A form for readers' comments may be provided at the back of this publication, or you may address your comments to the following address:

International Business Machines Corporation
Department 55JA, Mail Station P384
2455 South Road
Poughkeepsie, NY 12601-5400
United States of America

FAX (United States & Canada): 1+845+432+9405
FAX (Other Countries):
Your International Access Code +1+845+432-9405

IBMLink (United States customers only): IBMUSM10(MHVRCFS)
Internet e-mail: mhvrcfs@us.ibm.com
World Wide Web: http://www.ibm.com/servers/eserver/zseries/zos/webqs.html

If you would like a reply, be sure to include your name, address, telephone number, or FAX number.

Make sure to include the following in your comment or note:
- Title and order number of this book
- Page number or topic related to your comment

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

Contents

Figures

Tables

About This Book

This publication documents a new and enhanced version of SMP/E. New or changed information is identified by revision bars (|) to the left of the addition or change.

Who Should Read This Publication

Anyone who uses SMP/E, or who wants to understand SMP/E processes, should read this publication.

After reading this publication, you should be able to do most SMP/E processes. You may have to refer to *SMP/E Commands* for details on commands.

Bibliography

This section tells you more about the SMP/E library.

- The IBM SMP/E for z/OS and OS/390, V3R2 publications are available as printable PDF files and BookManager-viewable softcopy at
  `http://www.ibm.com/servers/eserver/zseries/zos/bkserv/`
- Table 1 lists the IBM SMP/E for z/OS and OS/390, V3R2 publications and briefly describes each one.
- For information on z/OS publications and more information on the IBM SMP/E for z/OS and OS/390, V3R2 books, see *z/OS Information Roadmap*.

Table 1. Publications for IBM SMP/E for z/OS and OS/390, V3R2

Title	Description
SMP/E Messages, Codes, and Diagnosis, GA22-7770	Explains SMP/E messages and return codes and the actions to take for each; and how to handle suspected SMP/E problems.
SMP/E Commands, SA22-7771	Explains SMP/E commands and processing in detail.
SMP/E Reference, SA22-7772	Explains SMP/E modification control statements, data sets, exit routines, and programming interfaces in detail and provides additional SMP/E reference material.
SMP/E User's Guide, SA22-7773	Describes how to use SMP/E to install programs and service.

Using LookAt to look up message explanations

LookAt is an online facility that lets you look up explanations for most messages you encounter, as well as for some system abends and codes. Using LookAt to find information is faster than a conventional search because in most cases LookAt goes directly to the message explanation.

You can access LookAt from the Internet at:
http://www.ibm.com/eserver/zseries/zos/bkserv/lookat/ or from anywhere in z/OS or z/OS.e where you can access a TSO/E command line (for example, TSO/E prompt, ISPF, z/OS UNIX System Services running OMVS).

The LookAt Web site also features a mobile edition of LookAt for devices such as Pocket PCs, Palm OS, or Linux-based handhelds. So, if you have a handheld device with wireless access and an Internet browser, you can now access LookAt message information from almost anywhere.

To use LookAt as a TSO/E command, you must have LookAt installed on your host system. You can obtain the LookAt code for TSO/E from a disk on your *z/OS Collection* (SK3T-4269) or from the LookAt Web site's **Download** link.

Summary of Changes

Summary of Changes
for SA22-7771-04
as Updated, August 2003

This revision reflects the deletion, addition, or modification of information to support miscellaneous maintenance items. A vertical bar (|) in the left margin indicates changes to the text and illustrations.

Summary of Changes
for SA22-7773-03
SMP/E Version 3 Release 2
May 2003

New Information

- "SMP/E V3R2 Overview" on page 172 has been added to "Appendix A. Migration". It includes migration information on:
  - "LINK LMODS Command" on page 172
  - "REPORT CALLLIBS Command Removal" on page 172
  - "UPGRADE Command" on page 172
  - "GIMXSID Service Routine" on page 173
  - "GIMZIP: Archive Segmentation" on page 173
  - "Java Archive Files" on page 174
  - "Smaller SMPLTS data set" on page 174
  - "DUMMY data set for SYSDEFSD" on page 175
  - "GIMZIP: User Defined Subdirectories" on page 173
  - "SMP/E Dialog Customization" on page 176
  - "GIMUTTBL Removal" on page 176.
- Chapter 18, "Java Archive Update Exploiter's Guide", on page 165 has been added.
- "Relinking Load Modules That Use CALLLIBS: LINK LMODS" on page 44 has been added.

Changed Information

- Member GIMSAMPU in SYS1.SAMPLIB has been updated to provide sample job steps to allocate SMPCSI data sets and SMP/E operational data sets (such as SMPPTS and SMPLOG) and UCLIN statements to initialize the newly allocated SMPCSI data sets, as shown in "Allocating a CSI Data Set" on page 52 and Table 24 on page 193.
- "Handling Cross-Zone Link-Edits: LINK MODULE" on page 43 has been updated.
- "How Dynamic Allocation Works" on page 60 has been updated.
- "Customize the SMP/E Dialogs" on page 70 has been updated.
- "Updating Target Libraries: APPLY" on page 115 has been updated.
- "How to Use LINK MODULE" on page 126 has been updated.
- "SMP/E Load Modules and Service Routines Moved to SYS1.MIGLIB" on page 180 has been updated.

- “GIMXTRX Service Routine” on page 180 has been updated.
- Appendix B, “Recommended Service Upgrade (RSU)”, on page 195 has been updated.

Moved Information

- None.

Deleted Information

- The chapter on using the REPORT CALLLIBS command has been deleted, because that command is no longer supported.

Summary of Changes
for SA22-7773-02
as Updated, March 2002

This revision reflects the deletion, addition, or modification of information to support miscellaneous maintenance items. A vertical bar (v) in the left margin indicates changes to the text and illustrations.

New Information

- An appendix with z/OS product accessibility information has been added.

Changed Information

- Table 24 on page 193 has been updated to include GIMCRSAM, GIMPRSAM, and GIMSAMPU.

Moved Information

- None.

Deleted Information

- None.

Summary of Changes
for SA22-7773-01
SMP/E Version 3
October 2001

The book contains information previously presented in SA22-7773-00, which applied to z/OS Version 1 Release 1.

New Information

- Appendix A, “Migration”, on page 169 has been added.
- SMPPARM member GIMDDALC in “How to Dynamically Allocate Data Sets to Be Used During SMP/E Processing” on page 58 has been added.

Changed Information

- “Example 1: Updating a Module” on page 152 has been updated.

Moved Information

- None.

Deleted Information

- Information related to backup IEANUC01 load modules has been removed.

This book contains terminology, maintenance, and editorial changes. Technical changes or additions to the text and illustrations are indicated by a vertical line to the left of the change. You may notice changes in the style and structure of some content in this book — for example, headings that use uppercase for the first letter of initial words only, and procedures that have a different look and format. The changes are ongoing improvements to the consistency and retrievability of information in our books.

Chapter 1. SMP/E Primer

This chapter provides an introduction to SMP/E to new SMP/E users. If you are already familiar with SMP/E, you can skip this chapter.

What Is SMP/E, and Why Should I Use It?

SMP/E is a tool designed to manage the installation of software products on your z/OS system and to track the modifications you make to those products. Usually, it is the system programmer's responsibility to ensure that all software products and their modifications are properly installed on the system. The system programmer also has to ensure that all products are installed at the proper level so all elements of the system can work together. At first, that might not sound too difficult, but as the complexity of the software configuration increases, so does the task of monitoring all the elements of the system. To better understand this, let's take a closer look at your z/OS system and see how SMP/E can help you maintain it.

Understanding Your System

Your z/OS system may appear to be one big block of code that drives your CPU. Actually, z/OS is a complex system comprising many different smaller blocks of code. Each of those smaller blocks of code perform a specific function within the system.

For example, some of the functions that can appear in a Z/OS system include:

- Base Control Program (BCP)
- C/C++ IBM Open Class Library
- Communications Server (CS z/OS)
- Cryptographic Services
- DCE Application Support
- DCE Base Services
- DFSMSdfp
- DFSORT
- Distributed File Service
- Encina Toolkit Executive
- Hardware Configuration Definition (HCD)
- High Level Assembler (HLASM)
- IBM HTTP Server
- Infoprint Server
- ISPF
- JES2 or JES3
- Language Environment
- Managed System Infrastructure (msys) for Setup
- Network File System
- Open Systems Adapter/Support Facility (OSA/SF)
- Resource Measurement Facility (RMF)
- System Display and Search Facility (SDSF)

- SMP/E
- Time Sharing Option/Extensions (TSO/E)
- z/OS UNIX® System Services (z/OS UNIX)

Each system function is composed of one or more load modules. In a z/OS environment, a load module represents the basic unit of machine-readable, executable code. Load modules are created by combining one or more object modules and processing them with a link-edit utility. The link-editing of modules is a process that resolves external references and addresses. The functions on your system, therefore, are one or more object modules that have been combined and link-edited.

To see where the object modules come from, let's take a look at the example in Figure 1.

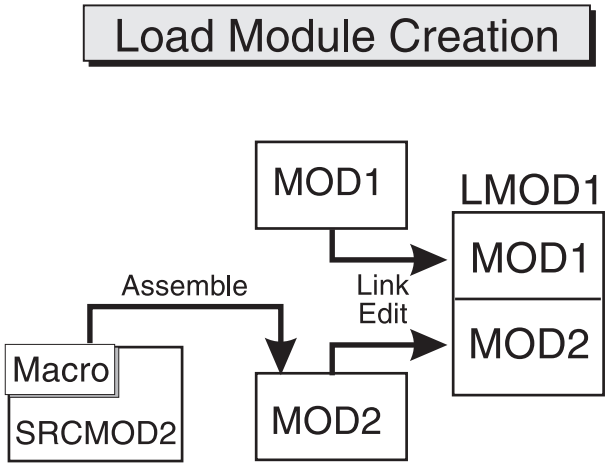


Figure 1. Load Module Creation

Most of the time, object modules are sent to you as part of a product. In this example, the object module MOD1 was sent as part of the product. Other times, you may need to assemble source code sent to you by product packagers to create the object module. You can modify the source code and then assemble it to produce an object module. In the example, SRCMOD2 is source code that you assemble to create object module MOD2. When assembled, you link-edit object module MOD2 with object module MOD1 to form the load module LMOD1.

In addition to object modules and source code, most products distribute many additional parts, such as macros, help panels, dialog elements, and other z/OS library members. These modules, macros, and other types of data and code are the basic building blocks of your system. All of these building blocks are called *elements*.

Changing the Elements of the System

Over time, you may need to change some of the elements of your system. These changes may be necessary to improve the usability or reliability of a product. You may want to add some new functions to your system, upgrade some of the elements of your system, or modify some elements for a variety of reasons. In all cases, you are making system modifications. In SMP/E, we refer to these system modifications as *SYSMODs*.

A SYSMOD is the actual package containing information SMP/E needs to install and track system modifications. SYSMODs are composed of two parts:

- Modification control statements (MCS), designated by **++** as the first two characters, that tell SMP/E:
  - What elements are being updated or replaced
  - How the SYSMOD relates to product software and other SYSMODs
  - Other specific installation information
- Modification text, which is the object modules, macros, and other elements supplied by the SYSMOD

There are four different categories of SYSMODs, each supporting a task you might want to perform:

Function SYSMODs
Introduce the elements for a product.

PTF (program temporary fix) SYSMODs
Prevent or fix problems with an element, or introduce new element s.

APAR (authorized program analysis reports) SYSMODs
Fix problems with an element.

USERMOD (user modifications) SYSMODs
Customize an element.

Introducing an Element—The Function SYSMOD

One way you can modify your system is to introduce new elements into that system. To accomplish this with SMP/E, you can install a function SYSMOD. The function SYSMOD introduces a new product, a new version or release of a product, or updated functions for an existing product into the system. All other types of SYSMODs are dependent upon the function SYSMOD, because they are all modifications of the elements originally introduced by the function SYSMOD.

When we refer to installing a function SYSMOD, we are referring to the placing of all the product's elements in the system data sets, or *libraries*. Examples of these libraries are SYS1.LPALIB, SYS1.MIGLIB, and SYS1.SVCLIB. Figure 2 depicts the process of creating executable code in the production system libraries.

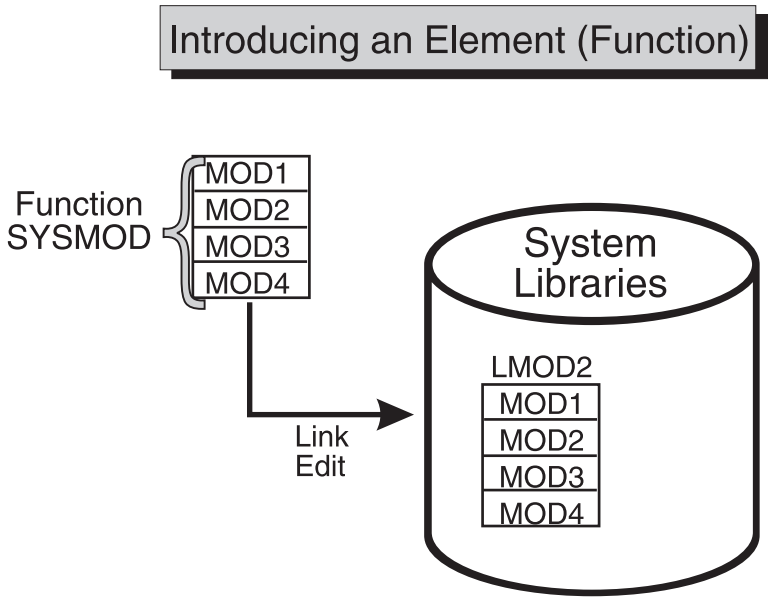


Figure 2. Introducing an Element

In this figure, the installation of a function SYSMOD link-edits object modules MOD1, MOD2, MOD3, and MOD4 to create load module LMOD2. The executable code created in load module LMOD2 is installed in the system libraries through the installation of the function SYSMOD.

There are two types of function SYSMODs:

- A *base* function SYSMOD adds or replaces an entire system function. Examples of base functions are SMP/E and JES2.
- A *dependent* function SYSMOD provides an addition to an existing system function. It is called dependent because its installation depends upon a base function already being installed. Examples of dependent functions are the language features for SMP/E.

Both base function SYSMODs and dependent function SYSMODs are used to introduce new elements into the system.

Here's an example of a simple function SYSMOD that introduces four elements:

```
++FUNCTION(FUN0001)         /* SYSMOD type and identifier.   */.
++VER(Z038)                 /* For MVS SREL                  */.
++MOD(MOD1)  RELFILE(1)     /* Introduce this module         */
            DISTLIB(AOSFB)  /* in this distribution library. */.
++MOD(MOD2)  RELFILE(1)     /* Introduce this module         */
            DISTLIB(AOSFB)  /* in this distribution library. */.
++MOD(MOD3)  RELFILE(1)     /* Introduce this module         */
            DISTLIB(AOSFB)  /* in this distribution library. */.
++MOD(MOD4)  RELFILE(1)     /* Introduce this module         */
            DISTLIB(AOSFB)  /* in this distribution library. */.
```

Preventing or Fixing Problems with an Element—The PTF SYSMOD

When a problem with a software element is discovered, IBM supplies its customers with a tested fix for that problem. This fix comes in the form of a program temporary fix (PTF). Although you may not have experienced the problem the PTF

is intended to prevent, it is wise to install the PTF on your system. The PTF SYSMOD is used to install the PTF, thereby preventing the occurrence of that problem on your system.

Usually, PTFs are designed to replace or update one or more complete elements of a system function. Let's look at Figure 3.

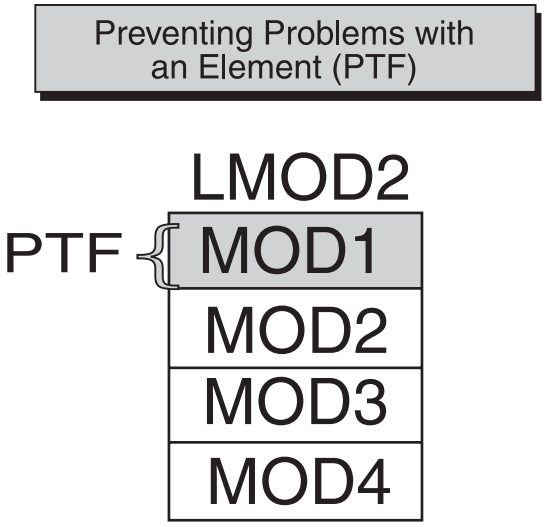


Figure 3. Preventing Problems with an Element

In Figure 3, we see a previously installed load module, LMOD2. If we want to replace the element MOD1, we should install a PTF SYSMOD that contains the module MOD1. That PTF SYSMOD replaces the element in error with the corrected element. As part of the installation of the PTF SYSMOD, SMP/E relinks LMOD2 to include the new and corrected version of MOD1.

Here is an example of a simple PTF SYSMOD:

```
++PTF(PTF0001)              /* SYSMOD type and identifier.   */.
++VER(Z038) FMID(FUN0001)   /* Apply to this product.        */.
++MOD(MOD1)                 /* Replace this module           */
            DISTLIB(AOSFB)  /* in this distribution library. */.
...
... object code for module
...
```

PTF SYSMODs are always dependent upon the installation of a function SYSMOD. In some cases, some PTF SYSMODs may also be dependent upon the installation of other PTF SYSMODs. These dependencies are called *prerequisites*. We will look at a typical PTF prerequisite when we discuss the complexity of keeping track of the elements of the system.

Fixing Problems with an Element—The APAR SYSMOD

You may sometimes find it is necessary to correct a serious problem that occurs on your system before a PTF is ready for distribution. In this situation, IBM supplies you with an authorized program analysis report (APAR). An APAR is a fix designed to quickly correct a specific area of an element or replace an element in error. You install an APAR SYSMOD to implement a fix, thereby updating the incorrect element.

In Figure 4, the shaded section shows an area of MOD2 containing an error.

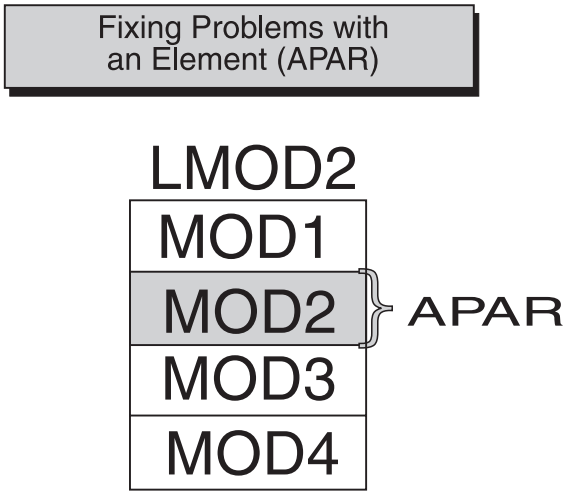


Figure 4. Fixing Problems with an Element

The processing of the APAR SYSMOD provides a modification for object module MOD2. During the installation of the APAR SYSMOD, MOD2 is updated (and corrected) in load module LMOD2.

Here is an example of a simple APAR SYSMOD:

```
++APAR(APAR001)            /* SYSMOD type and identifier.   */.
++VER(Z038) FMID(FUN0001)  /* Apply to this product         */
            PRE(UZ00004)   /* at this service level.        */.
++ZAP(MOD2)                /* Update this module            */
            DISTLIB(AOSFB) /* in this distribution library. */.
...
... zap control statements
...
```

The APAR SYSMOD always has the installation of a function SYSMOD as a prerequisite, and can also be dependent upon the installation of other PTF or APAR SYSMODs.

Customizing an Element—The USERMOD SYSMOD

If you had a requirement for a product to perform differently from the way it was designed, you might want to customize that element of your system. IBM provides you with certain modules that allow you to tailor IBM code to meet your specific needs. After making the desired changes, you add these modules to your system by installing a USERMOD SYSMOD. This SYSMOD can be used to replace or update an element, or to introduce a totally new user-written element into the system. In either case, the USERMOD SYSMOD is built by you either to change IBM code or to add your own code to the system.

In Figure 5, MOD3 has been updated through the installation of a USERMOD SYSMOD.

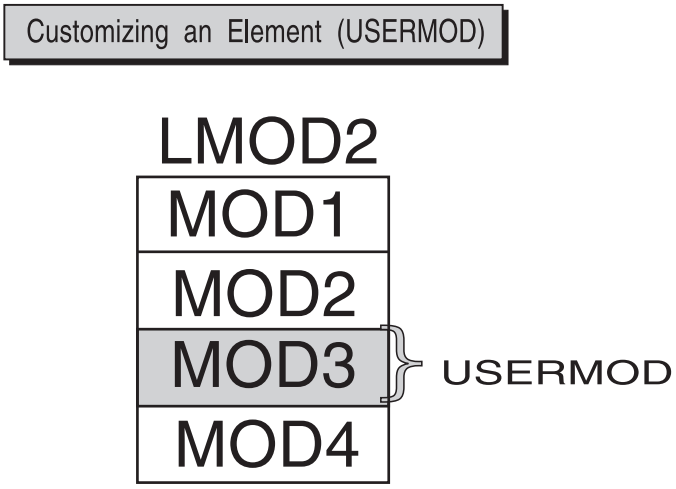


Figure 5. Customizing an Element

Here is an example of a simple USERMOD SYSMOD:

```
++USERMOD(USRMOD1)            /* SYSMOD type and identifier.   */.
++VER(Z038) FMID(FUN0001)     /* Apply to this product         */
            PRE(UZ00004)      /* at this service level.        */.
++SRCUPD(JESMOD3)             /* Update this source module     */
            DISTLIB(AOSFB)    /* in this distribution library. */.
...
... update control statements
...
```

Prerequisites for USERMOD SYSMODs are the installation of a function SYSMOD, and possibly the installation of other PTF, APAR, or USERMOD SYSMODs.

SYSMOD Prerequisites

As you have learned, PTF, APAR, and USERMOD SYSMODs all have the function SYSMOD as a prerequisite. In addition to their dependencies on the function SYSMOD:

- PTF SYSMODs may be dependent upon other PTF SYSMODs.
- APAR SYSMODs may be dependent upon PTF SYSMODs and other APAR SYSMODs.
- USERMOD SYSMODs may be dependent upon PTF SYSMODs, APAR SYSMODs, and other USERMOD SYSMODs.

Consider the complexity of these dependencies. When you multiply that complexity by hundreds of load modules in dozens of libraries, the need for a tool like SMP/E becomes apparent.

Let's examine the impact of these dependencies on the maintenance of software in a z/OS environment.

Keeping Track of the Elements of the System

The importance of keeping track of system elements and their modifications becomes readily apparent when we examine the z/OS maintenance process. Often, a PTF contains multiple element replacements. In the example in Figure 6, PTF1 contains replacements for two modules, MOD1 and MOD2. Although load module LMOD2 contains four modules, only two of those modules are being replaced.

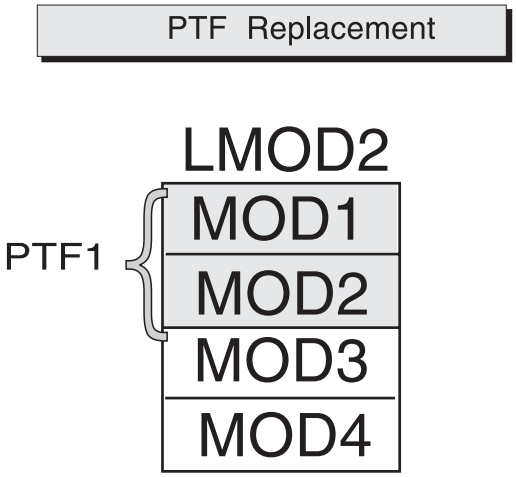


Figure 6. PTF Replacement

But what happens if a second PTF replaces some of the code in a module that was replaced by PTF1? Let's look at Figure 7.

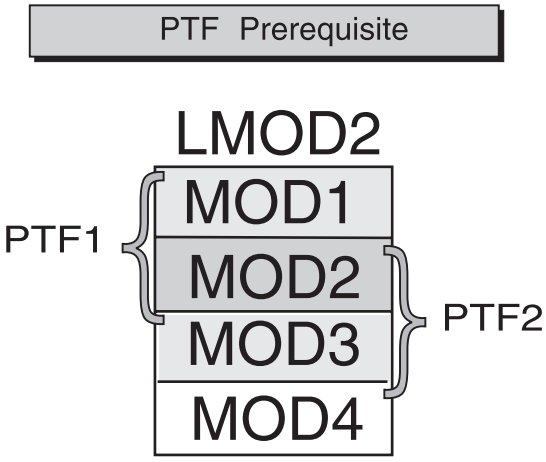


Figure 7. PTF Prerequisite

In this example, PTF2 contains replacements for MOD2 and MOD3. For MOD1, MOD2, and MOD3 to interface successfully, PTF1 must be installed before PTF2. That's because MOD3 supplied in PTF2 may depend on the PTF1 version of MOD1 to be present. It is this dependency that constitutes a prerequisite. SYSMOD prerequisites are identified in the modification control statements (MCS) part of the SYSMOD package we discussed in "Changing the Elements of the System" on page 2.

In addition to tracking prerequisites, there is another important reason to track system elements. The same module is often part of many different load modules. Let's take a look at the example in Figure 8 on page 9.

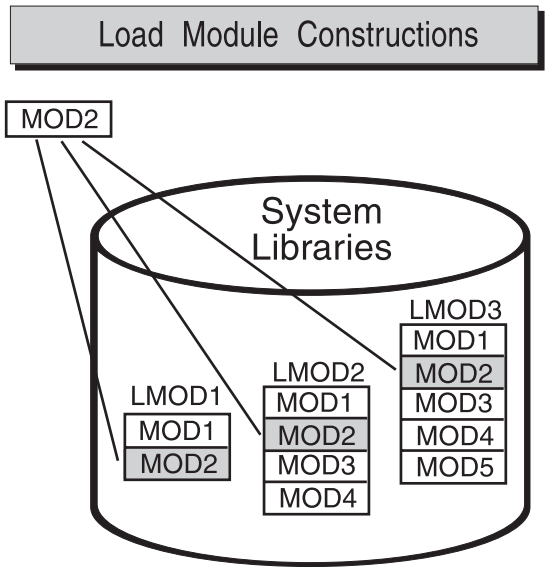


Figure 8. Load Module Constructions

In Figure 8, the same MOD2 module is present in LMOD1, LMOD2, and LMOD3. When a PTF is introduced that replaces the element MOD2, that module must be replaced in all the load modules in which it exists. Therefore, it is imperative that we keep track of all load modules and the modules they contain.

You can now appreciate how complicated the tracking of system elements and their modification levels can become. Let's take a brief look at how we implement the tracking capabilities of SMP/E.

Tracking and Controlling Requisites

To track and control elements successfully, all elements and their modifications and updates must be clearly identified to SMP/E. SMP/E relies on *modification identifiers* to accomplish this. There are three modification identifiers associated with each element:

- **Function modification identifiers (FMIDs)** that identify the function SYSMOD that introduced the element into the system.
- **Replacement modification identifiers (RMIDs)** that identify the last SYSMOD (usually a PTF SYSMOD) to replace the element.
- **Update modification identifiers (UMIDs)** that identify the SYSMODs that have updated an element since it was last replaced.

SMP/E uses these modification identifiers to track all SYSMODs installed on your system. This ensures that they are installed in the proper sequence. Now that we realize the need for element tracking and know the types of things SMP/E tracks, let's look at how SMP/E performs its tracking function.

How Does SMP/E Work?

Let's review our discussion of how functions are installed into the system. We begin with elements, such as modules, macros, and source code. These elements are then processed by utilities, such as an assembler or link-editor, to create load modules. The load modules contain the machine-readable, executable code.

Your production system in a z/OS environment consists of the z/OS operating system and all the code needed to do your everyday work. That's fine, but where is all that stuff kept, and how is it organized? Let's find out.

The Distribution and Target Libraries

To properly perform its processing, SMP/E must maintain a great deal of information about the structure, content, and modification status of the software it manages. Think of all the information SMP/E has to maintain as if it were all the information contained in the public library. To follow this analogy, let's refer to Figure 9.

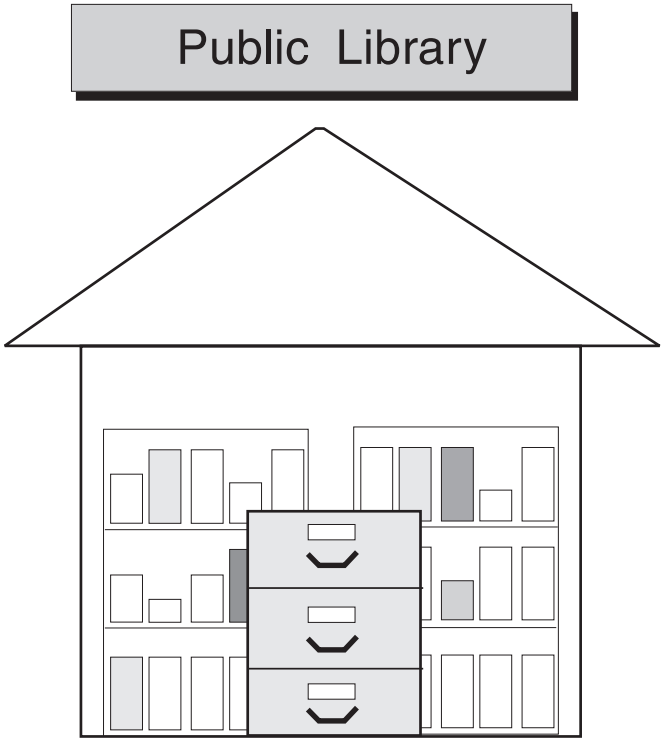


Figure 9. The Public Library

If you look at this figure depicting the public library, you see bookshelves filled with books and a card catalog with drawers containing a card for each book in the library. These cards contain information, such as the title, author, publishing dates, type of book, and a pointer to the actual book on the shelf.

In the SMP/E environment, there are two distinct types of "bookshelves." They are referred to as the *distribution libraries* and the *target libraries*. Figure 10 on page 11 depicts these two types of SMP/E libraries.

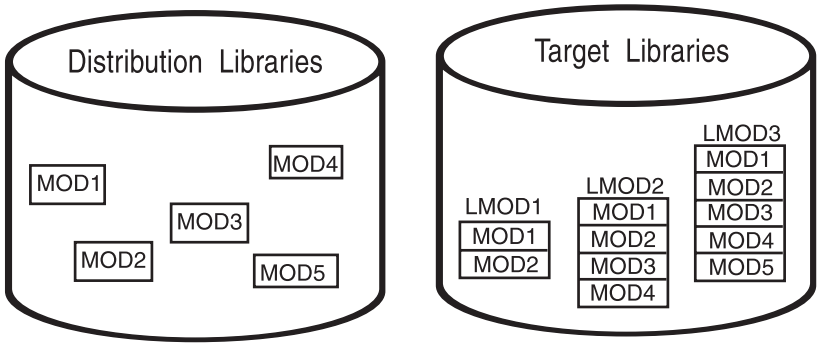


Figure 10. The Distribution and Target Libraries

In much the same way the bookshelves in the public library hold the library books, the distribution and target libraries hold the elements of the system.

Distribution libraries contain all the elements, such as modules and macros, that are used as input for running your system. One very important use of the distribution libraries is for backup. Should a serious error occur with an element on the production system, the element can be replaced by a stable level found in the distribution libraries.

Target libraries contain all the executable code needed to run the system.

The Consolidated Software Inventory (CSI)

As you refer to the analogy of the public library, you can see that there is one important piece of Figure 9 that we have not yet considered. In the public library, there is a card catalog to help you find the book or piece of information you are looking for. SMP/E provides the same type of tracking mechanism in the form of the *consolidated software inventory* (CSI).

The CSI data sets contain all the information SMP/E needs to track the distribution and target libraries. As the card catalog contains a card for each book in the library, the CSI contains an entry for each element in its libraries. The CSI entries contain the element name, type, history, how the element was introduced into the system, and a pointer to the element in the distribution and target libraries. The CSI does not contain the element itself, but rather a description of the element it represents.

Let's see exactly how these entries are arranged in the CSI.

The SMP/E Zones

The cards in the public library card catalog are arranged alphabetically by the author's last name, and by the topic and title of the book. In the CSI, entries for the elements in the distribution and target libraries are grouped according to their installation status. That is, entries representing elements found in the distribution libraries are contained in the *distribution zone*. Entries representing elements found in the target libraries are contained in the *target zone*. Both of these zones serve the same purpose as the drawers of the public library card catalog.

In addition to the distribution and target zones, the SMP/E CSI also contains a *global zone*. The global zone contains:

- Entries needed to identify and describe each target and distribution zone to SMP/E
- Information about SMP/E processing options
- Status information for all SYSMODs SMP/E has begun to process

- Exception data for SYSMODs requiring special handling or that are in error

In SMP/E, when we speak of exception data, we are usually referring to *HOLDDATA*. HOLDDATA is often supplied for a product to indicate a specified SYSMOD should be held from installation. Reasons for holding a SYSMOD can be:

- A PTF is in error and should not be installed until the error is corrected (ERROR HOLD).
- Certain system actions may be required before SYSMOD installation (SYSTEM HOLD).
- The user may want to perform some actions before installing the SYSMOD (USER HOLD).

All the information found in the global zone, combined with the information found in the distribution and target zones, represents the data SMP/E needs to install and track your system software.

Remember the picture of the public library in Figure 9 on page 10? Now look at Figure 11.

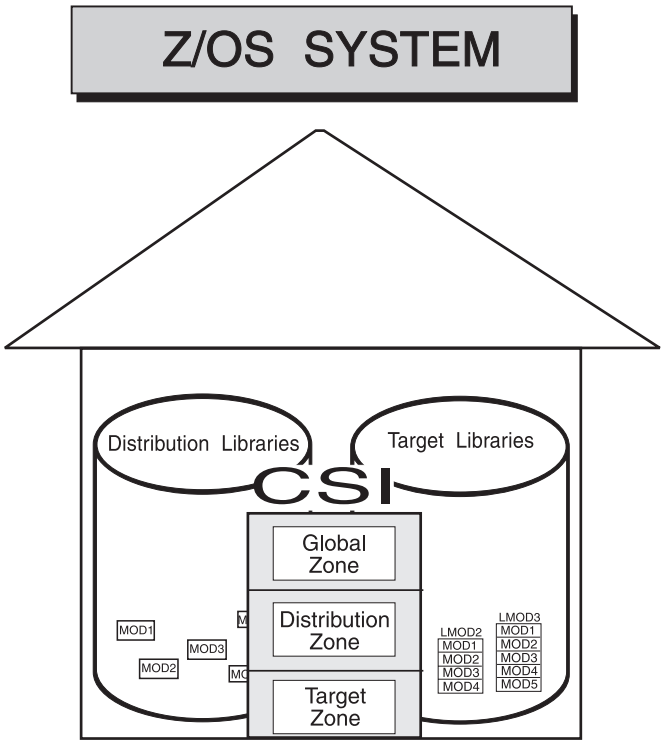


Figure 11. z/OS System with SMP/E

Now you can see how all the elements of the system fit together, and how they can be installed, modified, and tracked using SMP/E.

What Are the Basic SMP/E Commands I Need to Know?

Now that you are familiar with SMP/E and what it can do, you are probably wondering what you need to know to get started using SMP/E. Let's take a look at the basic processing commands you need to know to use SMP/E.

Setting the Zone You Want to Work On

Before processing SMP/E commands, you must first set the zone on which you want SMP/E to work (global, target, or distribution). You do this by issuing the **SET** command. The SET command identifies the zone and, therefore, the libraries, upon which subsequent SMP/E commands are to act.

The SET command can also be used to request a particular set of predefined processing options. For more information about the SET command, refer to *SMP/E Commands*.

Receiving the SYSMOD into SMP/E's Data Sets

For SMP/E to install a SYSMOD, the SYSMOD must be "received" into data sets that can be used by SMP/E. The SMP/E RECEIVE command performs the task of copying the SYSMOD from the distribution medium from which it was sent into the data sets used by SMP/E.

For more information about the RECEIVE command, refer to "Receiving the SYSMOD into SMP/E's Data Sets" on page 14.

Applying the SYSMOD to the Target Libraries

Once a SYSMOD has been received, you want to "apply" the SYSMOD to the appropriate target libraries. The SMP/E APPLY command invokes various system utilities to install the SYSMOD's elements into the target libraries.

For more information about the APPLY command, refer to "Applying the SYSMOD to the Target Libraries" on page 18.

Restoring the Target Libraries to the Previous Level

Should you experience problems after applying a SYSMOD, you may want to "restore" its elements in error to a previous and stable level. The SMP/E RESTORE command replaces a failing element with a copy from the distribution libraries.

For more information about the RESTORE command, refer to "Restoring the Target Libraries to the Previous Level" on page 22.

Accepting the SYSMOD and Updating the Distribution Libraries

After you have performed a SYSMOD RECEIVE and APPLY, you want to "accept" the elements into the distribution libraries for backup. However, this should be done only after you are satisfied with the performance and stability of the elements of the SYSMOD. Once you ACCEPT a SYSMOD, you cannot RESTORE its element to a previous level. The SMP/E ACCEPT command updates the distribution libraries so they are available for backup of any future SYSMODs.

For more information about the ACCEPT command, refer to "Accepting the SYSMOD into the Distribution Libraries" on page 25.

Displaying SMP/E Data

The SMP/E CSI and other primary data sets contain a great deal of information you may find useful when installing new elements or functions, preparing user modifications, or debugging problems. There are several ways SMP/E allows you to display that information, as well as information about modules, macros, and other elements:

- **Query dialogs** display specific information you request through interactive dialogs with SMP/E.
- The **LIST command** generates a hardcopy listing of information about your system.
- **REPORT commands** check, compare, and generate hardcopy information about the contents of zones on your system.
- The **SMP/E CSI application programming interface** can be used to write application programs to query the contents of your system's CSI data sets.

For more information about displaying SMP/E data, refer to "Displaying SMP/E Data" on page 29.

Flow of SMP/E SYSMOD Processing

To see the flow of SMP/E SYSMOD processing for the RECEIVE, APPLY, RESTORE, and ACCEPT commands, let's look at Figure 12.

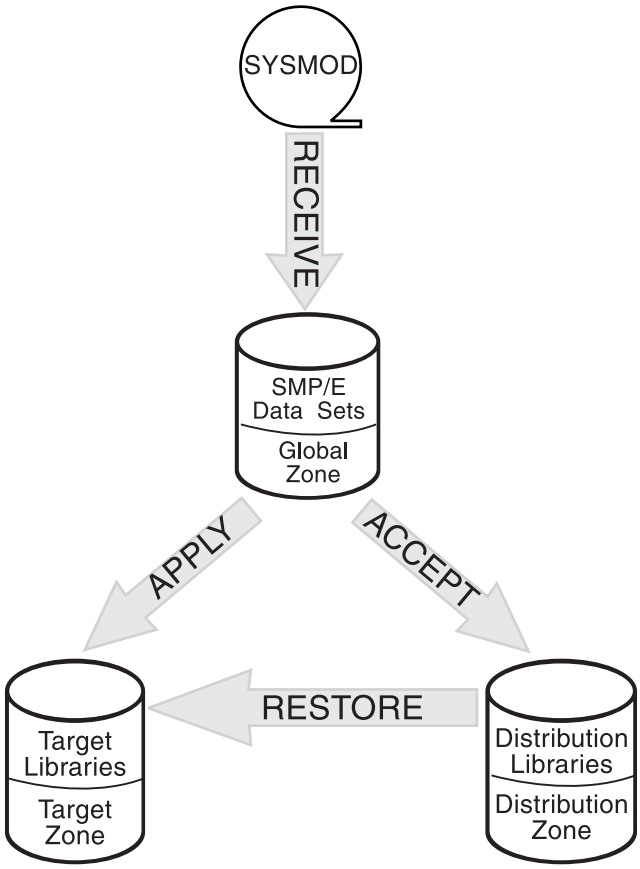


Figure 12. Flow of SMP/E SYSMOD Processing

Receiving the SYSMOD into SMP/E's Data Sets

To initiate SMP/E processing, you must first install the software into SMP/E data sets. You can use the RECEIVE command to load the SYSMOD information from the distribution medium into the SMPPTS and SMPTLIB data sets for later installation of the SYSMODs.

In this chapter, you will learn about those data sets and the following topics:
- What happens during RECEIVE processing

- How SMP/E keeps track of RECEIVE processing
- Examples of using the RECEIVE command
- A summary of the RECEIVE command

What Happens During RECEIVE Processing

SMP/E knows software in terms of SYSMODs. Each SYSMOD processed by SMP/E contains two types of information:

- Instructions telling SMP/E what elements are in the SYSMOD and how to install them
- The actual element replacements or updates contained in the SYSMOD

The instructions are made up of a series of control statements called *modification control statements* (MCSs). The element replacements or updates can be packaged in several ways:

- The **RELFILE** method packages the elements in relative files that are separate from the MCSs. This method is used mostly for function SYSMODs. (The examples in the remainder of this book assume that function SYSMODs are packaged in RELFILE format.)
- The **inline** method packages the elements immediately following the associated MCSs.
- The **indirect library** method packages elements in DASD data sets that are separate from the MCSs.

For more details about packaging, see the *z/OS Packaging Rules* manual.

During RECEIVE processing, the MCS for each SYSMOD is copied to an SMP/E temporary storage area called the *SMPPTS data set*. The MCS entry contains the MCS and any inline element replacements or updates for the SYSMOD. Relative files, however, are stored in another temporary storage area called the *SMPTLIB data sets*.

We briefly mentioned HOLDDATA earlier in the book (see "The SMP/E Zones" on page 11). HOLDDATA is processed by the RECEIVE command and is stored for use later on during installation of the affected SYSMODs.

How SMP/E Keeps Track of RECEIVE Processing

SMP/E updates the global zone with information about the SYSMODs that have been received:

- **SYSMOD entries** are created in the global zone for each SYSMOD that has been received.
- **HOLDDATA entries** are created in the global zone for each ++HOLD statement that has been received. HOLDDATA entries identify SYSMODs that should be held back from being installed because they require special handling or are in error.

Figure 13 shows what you have learned about RECEIVE processing.

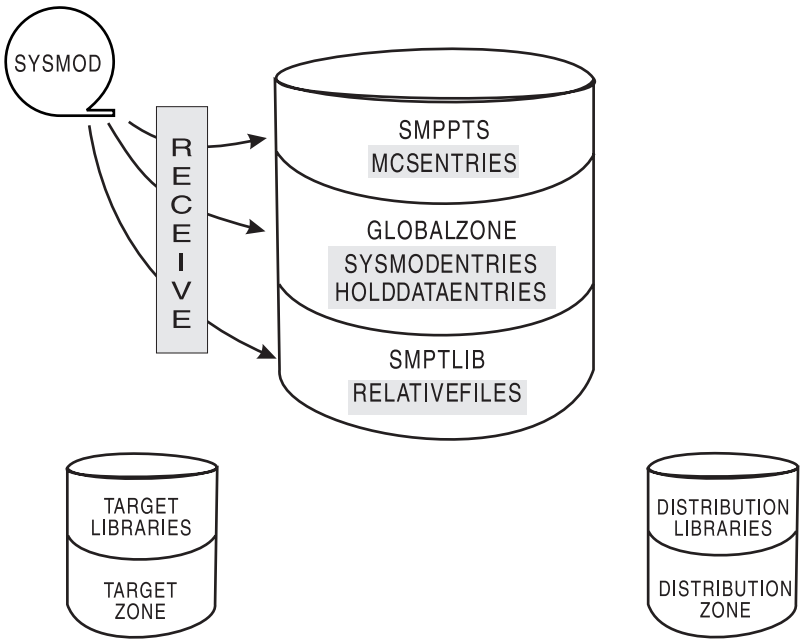


Figure 13. Results of RECEIVE Processing

Using the RECEIVE Command

In this section, you will see some basic examples of how you might use the RECEIVE command.

Examples

Let's look at a few of these examples.

Receiving SYSMODs and HOLDDATA: In the course of maintaining your system, you need to install service and process the related HOLDDATA. Assume IBM has supplied you with a service tape (such as a CBPDO tape or an ESO tape), and you want to install it on your system. The first step is to receive the SYSMODs and HOLDDATA that are contained on the tape. You can accomplish this by specifying the following commands:

```
SET      BDY(GLOBAL).
RECEIVE.
```

When you issue these commands, SMP/E receives all the SYSMODs and HOLDDATA on the service tape into the global zone.

Receiving Only HOLDDATA: There may be times when you do not want to receive the SYSMODs from a service tape, but you do want to receive the HOLDDATA. Because the HOLDDATA provides information about SYSMODs requiring special handling or that are in error, it is important for you to receive the HOLDDATA into SMP/E's storage repository as soon as possible. The following commands process only the HOLDDATA:

```
SET      BDY(GLOBAL).
RECEIVE  HOLDDATA.
```

By issuing these commands, you direct SMP/E to receive only the HOLDDATA from the service tape into the global zone.

Receiving Only SYSMODs: Assume you have previously received only the HOLDDATA from a service tape and are now ready to install the SYSMODs. To install these SYSMODs (using the APPLY and ACCEPT commands), you must first receive them. This can be done by specifying the following commands:

```
SET     BDY(GLOBAL).
RECEIVE SYSMODS.
```

When you issue these commands, you direct SMP/E to receive only the SYSMODs from the service tape into the global zone.

Receiving SYSMODs and HOLDDATA for a Specific Product: You may want to receive SYSMODs and HOLDDATA for a particular product from the service tape. You can accomplish this task by specifying the following commands:

```
SET     BDY(GLOBAL).
RECEIVE FORFMID(HOP0001).
```

By issuing these commands, you direct SMP/E to receive SYSMODs and HOLDDATA for the product whose FMID is HOP0001 from the service tape into the global zone.

For a more complete description of all the RECEIVE command operands and other examples, see The RECEIVE Command in *SMP/E Commands*.

Reporting Output

When RECEIVE processing is complete, these reports will help you analyze the results:

- The **RECEIVE Summary report** provides you with an at-a-glance look at all the SYSMODs that were processed during the RECEIVE command run. It shows you which SYSMODs were received, which were not received, and why.

  Note: The SYSMODs listed in this report depend on the operands you specify on the RECEIVE command.
- The **RECEIVE Exception SYSMOD Data report** provides you with a quick summary of the HOLDDATA information processed during the RECEIVE command run. It lists the SYSMODs requiring special handling or that are in error, and those SYSMODs no longer requiring special handling or that have had an error fixed.
- The **File Allocation report** provides you with a list of the data sets used for RECEIVE processing and supplies information about these data sets.

For more information about these reports (and samples of actual reports), see SMP/E Reports in *SMP/E Commands*.

Summary

Let's summarize what you have learned about using the RECEIVE command to load a SYSMOD into SMP/E's storage area. The RECEIVE command:

- Copies the MCS for each SYSMOD to the SMPPTS data set
- Loads elements into SMPTLIB data sets for SYSMODs using the relative file packaging method
- Records what is received in the global zone
  - SYSMOD entries
  - HOLDDATA entries
- Reports the results of processing

Applying the SYSMOD to the Target Libraries

After the SYSMODs have been received, you can use the APPLY command to install them into the appropriate target system libraries. The APPLY command calls system utilities, which are responsible for the actual updating of those libraries.

In this chapter, you will learn about the following topics:
- What happens during APPLY processing
- How SMP/E keeps track of APPLY processing
- Examples of using the APPLY command
- A summary of the APPLY command

What Happens During APPLY Processing

Throughout the APPLY process, SMP/E helps you manage the complexities of your system when installing SYSMODs.

Selecting the SYSMODs

You can specify operands on the APPLY command that tell SMP/E which of the received SYSMODs are to be selected for installation in the target libraries. SMP/E checks to make sure all other required SYSMODs (prerequisites) have been installed or are being installed concurrently and in the proper sequence. For more information about prerequisites, see "Keeping Track of the Elements of the System" on page 7.

Selecting the Elements

During APPLY processing, SMP/E uses the information provided in the selected SYSMODs to determine which elements should be installed in the target libraries. The selection of elements is monitored by SMP/E to make sure that the correct functional level of each element is selected.

Checking the APPLY Process

SMP/E provides you with an option to stop APPLY processing just before any updating takes place so you can ensure all prerequisites are satisfied before the installation of the SYSMODs. This helps you see what will happen (and helps you detect problem SYSMODs) without actually updating the target libraries.

Updating the Target Libraries

After the proper SYSMODs have been selected and the proper functional and service level of each element has been determined, the APPLY command directs SMP/E to call the system utilities. It is the system utilities that actually place the elements into the target libraries described in the target zone. The source of the elements is the SMPTLIB data sets, the SMPPTS data set, or the indirect libraries, depending on how the SYSMOD was packaged.

Note: Because the APPLY command updates the system libraries, you should **never** use it on a live production system. When you process the APPLY command, you should always use a **copy** of the target libraries and target zone. By using a copy, you minimize the risk of new code causing an outage of your system. This process of copying is called *cloning* and is explained in detail in the *OS/390 Software Management Cookbook*, SG24-4775.

How SMP/E Keeps Track of APPLY Processing

SMP/E updates the information about the SYSMODs that have been applied. Remember, the target zone reflects the contents of the target libraries. Therefore, after the utility work is complete, and the target libraries have been updated, the target zone is updated to accurately reflect the status of those libraries.

- A SYSMOD entry is created in the **target zone** for each SYSMOD that has been applied. Element entries (such as MOD and LMOD) are also created in the target zone for those elements that have been installed in the target libraries.
- SYSMOD entries in the **global zone** are updated to reflect that the SYSMOD has been applied to the target zone.
- BACKUP entries are created in the **SMPSCDS data set** so the SYSMOD can later be restored, if necessary.

Figure 14 shows what you have learned about APPLY processing.

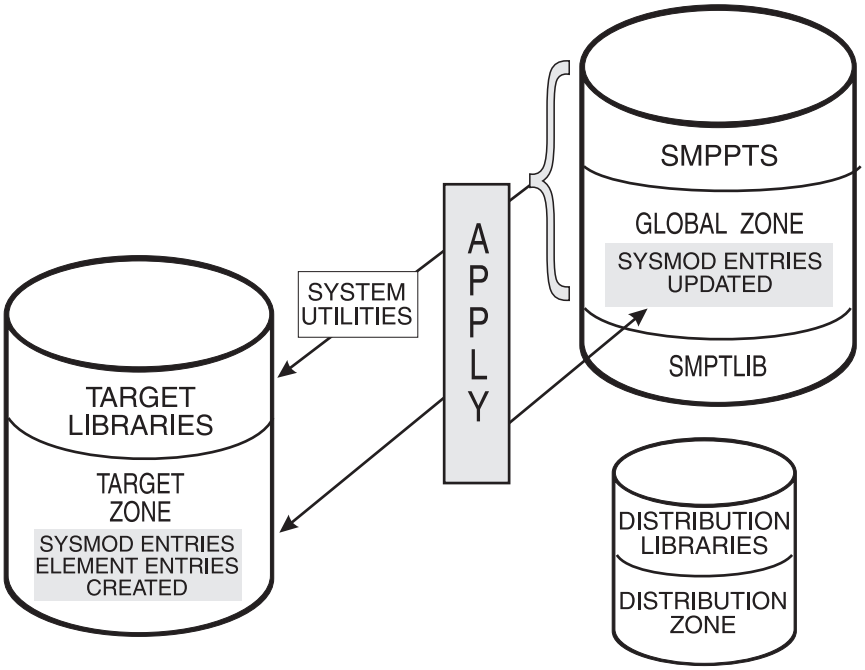


Figure 14. Results of APPLY Processing

Using the APPLY Command

The APPLY command has many operands that allow you great flexibility in choosing which SYSMODs you want installed in your target libraries. It also provides you with a variety of output based on the operands you specify.

Examples

Let's look at a few examples of how you might use the APPLY command.

Applying PTF SYSMODs: After you have received the SYSMODs into the global zone, you can tell SMP/E that you want to install only the PTF SYSMODs. You can do this by specifying the following commands:

```
SET      BDY(ZOSTGT1).
APPLY    PTFS.
```

By issuing these commands, you direct SMP/E to apply all eligible PTF SYSMODs to target zone ZOSTGT1.

Suppose you do not want to install all the PTF SYSMODs, but only a select few. You can do this by specifying the following commands:

```
SET      BDY(ZOSTGT1).
APPLY    SELECT(UZ00001,UZ00002).
```

Issuing these commands results in the selection of only PTFs UZ00001 and UZ00002 for installation in target zone ZOSTGT1.

Applying APAR and USERMOD SYSMODs: You may want to install just corrective fixes (APARs) or user modifications (USERMODs) into the target libraries. You can accomplish this with the following commands:

```
SET      BDY(ZOSTGT1).
APPLY    APARS
         USERMODS.
```

When you issue these commands, SMP/E installs all eligible APARs and USERMODs into target zone ZOSTGT1.

Applying SYSMODs for Selected Products: There may be times when you want to update only certain products on your system with the SYSMODs contained on a service tape. Assume you want to install all PTFs for a particular product to your system. This can be accomplished by specifying the following commands:

```
SET      BDY(ZOSTGT1).
APPLY    PTFS
         FORFMID(HOP0001).
```

or

```
SET      BDY(ZOSTGT1).
APPLY    FORFMID(HOP0001).
```

In both cases, SMP/E applies all applicable PTFs for the product with FMID HOP0001 to target zone ZOSTGT1. Unless you specify otherwise, PTFS is the default SYSMOD type.

Applying SYSMODs Having Prerequisites: When installing a SYSMOD, you might not always know if it has prerequisites, or if the prerequisites are available. (Sometimes a prerequisite SYSMOD might not be received, or it might be held because it is in error.) In cases such as this, you can direct SMP/E to check whether an equivalent (or superseding) SYSMOD is available, by specifying the GROUPEXTEND operand.

Assume you want to update a product with all the eligible PTFs and APARs. You can do this by specifying the following commands:

```
SET      BDY(ZOSTGT1).
APPLY    PTFS
         APARS
         FORFMID(HOP0001)
         GROUPEXTEND.
```

By issuing these commands, you direct SMP/E to apply all PTFs and APARs, along with any other required SYSMODs to the product whose FMID is HOP0001 and is located in the ZOSTGT1 target zone. If SMP/E cannot find a required SYSMOD, it looks for and uses a SYSMOD that supersedes the required one.

Applying SYSMODs Using the CHECK Operand: In the previous example, you directed SMP/E to automatically include all SYSMODs needed for the specified product. There may be times when you want to see which SYSMODs are included before you actually install them. You can do this with the CHECK operand by issuing the following commands:

```
SET     BDY(ZOSTGT1).
APPLY   PTFS
        APARS
        FORFMID(HOP0001)
        GROUPEXTEND
        CHECK.
```

After these commands are processed, you can check the SYSMOD Status report to see which SYSMODs would have been installed if you had not specified the CHECK operand. If you are satisfied with the results of this trial run, you can issue the commands again, without the CHECK operand, to actually install the SYSMODs.

For a more complete description of all the APPLY command operands, and for additional examples, see APPLY Command in *SMP/E Commands*.

Reporting Output

When APPLY processing is complete, these reports will help you analyze the results:

- The **SYSMOD Status report** provides you with a summary of the processing that took place for each eligible SYSMOD, based on the operands you specified on the APPLY command. It shows you which SYSMODs were applied, which were not applied, and why.
- The **Element Summary report** provides you with a detailed look at each element affected by APPLY processing. It tells you in which libraries the elements were installed.
- The **Causer SYSMOD Summary report** provides you with a list of SYSMODs that caused other SYSMODs to fail, and describes the errors that must be fixed to successfully process the SYSMODs. This report can reduce the amount of work involved in figuring out which errors caused SYSMODs to fail.
- The **File Allocation report** provides you with a list of the data sets used for APPLY processing and supplies information about these data sets.

Additional reports may be produced depending on the work being done and the content of the SYSMODs. For more information about all the reports produced by the APPLY command (and samples of actual reports), see The APPLY Command in *SMP/E Commands*.

Summary

Let's summarize what you have learned about using the APPLY command to install a SYSMOD in the target libraries. The APPLY command:

- Selects SYSMODs to install
- Checks that all other required SYSMODs have been (or are being) installed
- Based on SYSMODs, selects elements to install
- Directs SMP/E to call the system utilities to update the target libraries
- Records what is applied:
  - Target zone: Creates SYSMOD entries and element entries
  - Global zone: Updates SYSMOD entries
  - SMPSCDS data set: Creates BACKUP entries
- Reports the results of processing

Remember, you should **never** perform APPLY processing on a live production system!

Restoring the Target Libraries to the Previous Level

If you discover that a particular SYSMOD is causing a problem in your target libraries, you can remove it and replace the elements affected by it with the previous level of those elements, which is obtained from the backup (or distribution) libraries. If you are wondering how a backup version came to exist in the distribution libraries, this topic is covered in "Accepting the SYSMOD into the Distribution Libraries" on page 25.

You can use the RESTORE command to remove SYSMODs from the target libraries and restore them to a previous level. The RESTORE command reverses APPLY processing, but has no effect on ACCEPT processing.

In this chapter, you will learn about the following topics:
- What happens during RESTORE processing
- How SMP/E keeps track of RESTORE processing
- Examples of using the RESTORE command
- A summary of the RESTORE command

What Happens During RESTORE Processing

SMP/E provides you with a method for removing an applied SYSMOD when its installation results in unexpected problems.

Removing the SYSMODs

SMP/E ensures the eligibility of the selected SYSMODs and checks whether other SYSMODs are affected before continuing with RESTORE processing. Because of the various relationships and dependencies among the many SYSMODs, this checking is very important to the integrity of your system. In fact, to ensure that the requisites for a SYSMOD being restored are processed appropriately, SMP/E may require the whole chain of prerequisites to be restored.

Selecting the Elements

During RESTORE processing, SMP/E uses the information provided in the selected SYSMODs to determine which elements in the target zone should be replaced by elements in the related distribution libraries. The selection of elements is monitored by SMP/E to make sure that the correct functional level of each element is selected.

Checking the RESTORE Process

SMP/E provides you with an option to stop RESTORE processing just before any updating takes place so you can ensure all prerequisites are satisfied before restoring any SYSMODs. This helps you see what will happen without actually making any changes to the elements in the target libraries.

Replacing the Elements in the Target Libraries

When SMP/E is satisfied that the proper SYSMODs have been selected, it uses information from the target zone to determine which distribution zone describes the elements necessary to replace the SYSMOD's elements in the target libraries. The RESTORE command directs SMP/E to call system utilities that replace the elements in the target libraries with the previous level of the elements from the related distribution libraries.

How SMP/E Keeps Track of RESTORE Processing

SMP/E updates the information about the SYSMODs that have been restored. Remember, the target zone reflects the contents of the target libraries. Therefore,

after the utility work is complete, and the target libraries have been updated, the target zone is updated to accurately reflect the status of those libraries.

- All information in the **target zone** pertaining to the restored SYSMOD is removed. The element entries in the target zone are restored to reflect the distribution zone level of the elements.
- The **global zone** SYSMOD entries and MCS statements, which are stored in the SMPPTS data set, are deleted for those SYSMODs that have been restored. Any SMPTLIB data sets created during RECEIVE processing are also deleted for the restored SYSMOD. SMP/E automatically performs this global zone clean-up, unless you specify otherwise.

Figure 15 shows what you have learned about RESTORE processing.

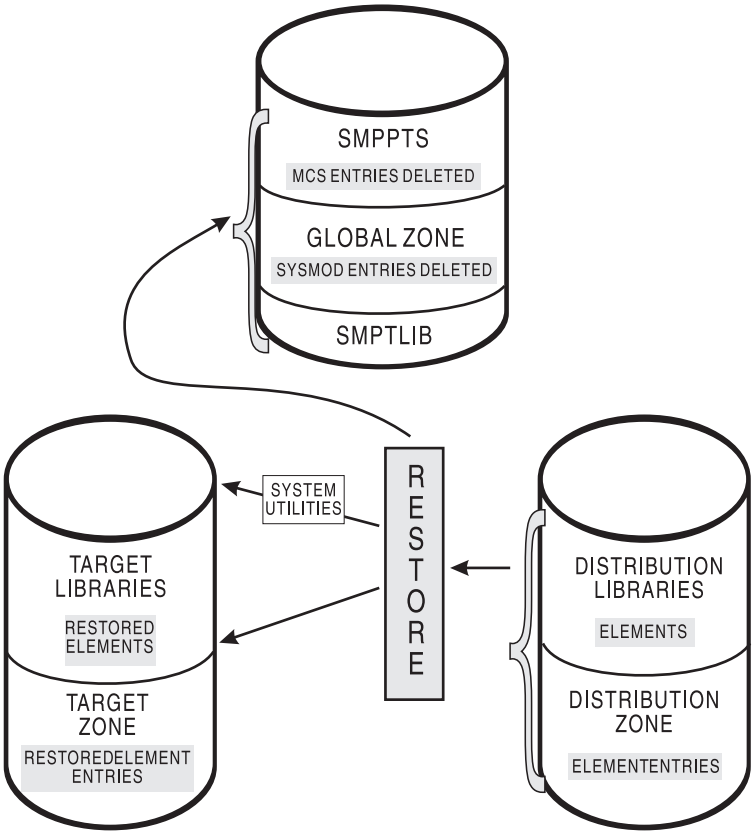


Figure 15. Results of RESTORE Processing

Using the RESTORE Command

The RESTORE command has operands that allow you to specify the criteria for removing SYSMODs from the target libraries. It also produces output that reports on its processing.

Examples

Let's look at a few examples of how you might use the RESTORE command.

Restoring a Single SYSMOD: Assume you have applied a SYSMOD and, after some initial testing, you discover that a PTF SYSMOD is causing problems on your system. You can remove this SYSMOD by specifying the following commands:

```
SET     BDY(ZOZTGT1).
RESTORE SELECT(UZ00001).
```

By issuing these commands, you instruct SMP/E to remove PTF UZ00001 from target zone ZOZTGT1 and replace its elements in the target libraries with the previous level of elements from the distribution libraries.

Restoring SYSMODs Using the GROUP Operand: When you want to remove a particular SYSMOD, it is not always easy to determine other SYSMODs that need to be restored in order to remove the bad one. Assume a particular PTF SYSMOD is causing a problem, and you want to know if it is dependent on any other SYSMODs so you can also restore those SYSMODs. This can be accomplished by specifying the following commands:

```
SET     BDY(ZOZTGT1).
RESTORE SELECT(UZ00003)
        GROUP.
```

By issuing these commands, you instruct SMP/E to restore PTF UZ00003 and any other related PTFs from target zone ZOZTGT1, and replace their elements with the previous level from the distribution zone.

Restoring SYSMODs Using the CHECK Operand: In the previous example, you directed SMP/E to restore any dependent SYSMODs in order to remove the bad one. There may be times when you want to see which SYSMODs are restored without actually restoring them. You can do this with the CHECK operand by issuing the following commands:

```
SET     BDY(ZOZTGT1).
RESTORE SELECT(UZ00003)
        GROUP
        CHECK.
```

After these commands are processed, you can check the SYSMOD Status report to see which SYSMODs would have been restored if you had not specified the CHECK operand. If you are satisfied with the results of this trial run, you can issue the commands again, without the CHECK operand, to actually restore the SYSMODs.

For a more complete description of all the RESTORE command operands, and for additional examples, see The RESTORE Command in *SMP/E Commands*.

Reporting Output

When RESTORE processing is complete, these reports will help you analyze the results:

- The **SYSMOD Status report** provides you with a summary of the processing that took place for each eligible SYSMOD, based on the operands you specified on the RESTORE command. It shows you which SYSMODs were restored, which were not restored, and why.
- The **Element Summary report** provides you with a detailed look at each element replaced or modified by RESTORE processing. It tells you in which libraries the elements were restored.
- The **Causer SYSMOD Summary report** provides you with a list of SYSMODs that caused other SYSMODs to fail, and describes the errors that must be fixed to successfully process the SYSMODs. This report can reduce the amount of work involved in figuring out which errors caused SYSMODs to fail.
- The **File Allocation report** provides you with a list of the data sets used for RESTORE processing and supplies information about these data sets.

Additional reports may be produced depending on the work being done and the content of the SYSMODs. For more information about all the reports produced by the RESTORE command (and samples of actual reports), see The RESTORE Command in *SMP/E Commands*.

Summary

Let's summarize what you have learned about using the RESTORE command to remove a SYSMOD from the target libraries. The RESTORE command:

- Removes the SYSMOD from the indicated target zone
- Calls system utilities to replace the SYSMOD's elements in the target libraries with elements from the related distribution libraries
- Records what is restored:
  - Target zone: Restores element entries to reflect their distribution zone level and deletes all information about restored SYSMOD.
  - Global zone: Deletes SYSMOD entries and MCS statements in SMPPTS for restored SYSMOD. Any SMPTLIB data sets created during RECEIVE processing are also deleted for the restored SYSMOD. (This global zone processing is optional.)
  - SMPSCDS data set: Deletes BACKUP entries for restored SYSMOD.
- Reports the results of processing

Note: Not all SYSMODs can be restored. For example, SMP/E cannot restore a SYSMOD that deletes another SYSMOD or that deletes a load module during APPLY processing.

Accepting the SYSMOD into the Distribution Libraries

You can use the ACCEPT command to install software in backup (or distribution) libraries. ACCEPT processing is very similar to APPLY processing with one important exception: ACCEPT processing is irreversible.

In this chapter, you will learn about the following topics:
- What happens during ACCEPT processing
- How SMP/E keeps track of ACCEPT processing
- Examples of using the ACCEPT command
- A summary of the ACCEPT command

What Happens During ACCEPT Processing

After you are satisfied that an applied SYSMOD has performed reliably in your target system, you can install it in your backup system (distribution) libraries.

Selecting the SYSMODs

You can specify operands on the ACCEPT command that tell SMP/E which of the received SYSMODs are to be selected for installation in the distribution libraries. SMP/E ensures that all other required SYSMODs have been installed or are being installed concurrently and in the proper sequence.

Selecting the Elements

During ACCEPT processing, SMP/E uses the information provided in the selected SYSMODs to determine which elements should be installed in the distribution libraries. The selection of elements is monitored by SMP/E to make sure that the correct functional level of each element is selected.

Checking the ACCEPT Process

SMP/E provides you with an option to stop ACCEPT processing before any updating takes place so you can ensure all prerequisites are satisfied before the installation of the SYSMODs. This helps you see what will happen (and helps you detect problem SYSMODs) without actually updating the distribution libraries.

Updating the Distribution Libraries

After the proper SYSMODs have been selected and the proper functional and service level of each element has been checked, SMP/E calls the system utilities (in the same manner as APPLY and RESTORE) to place the elements into the distribution libraries described in the distribution zone. The source of the elements is the SMPTLIB data sets, the SMPPTS data set, or the indirect libraries, depending on how the SYSMOD was packaged.

Note: When ACCEPT processing has been completed, there is no way it can be undone.

How SMP/E Keeps Track of ACCEPT Processing

SMP/E updates the information about the SYSMODs that have been accepted. Remember, the distribution zone reflects the contents of the distribution libraries. Therefore, after the utility work is complete, and the distribution libraries have been updated, the distribution zone is updated to accurately reflect the status of those libraries.

- A SYSMOD entry is created in the **distribution zone** for each SYSMOD that has been accepted. Element entries (such as MOD and LMOD) are also created in the distribution zone for the elements that have been installed in the distribution libraries.
- **Global zone** SYSMOD entries and MCS statements in the SMPPTS data set are deleted for those SYSMODs that have been accepted into the distribution zone. Any SMPTLIB data sets created during RECEIVE processing are also deleted. If you do not want SMP/E to do this global zone clean-up, you have the option to indicate this to SMP/E, and the information is saved.

Figure 16 on page 27 shows what you have learned about ACCEPT processing.

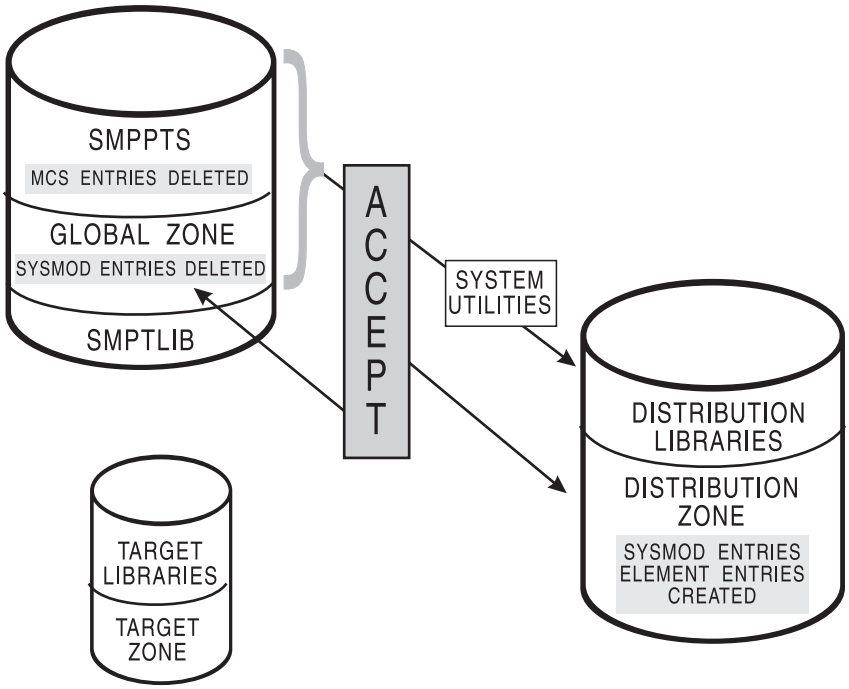


Figure 16. Results of ACCEPT Processing

Using the ACCEPT Command

The ACCEPT command has many operands that allow you great flexibility for further defining which SYSMODs you want installed in your distribution libraries. It also provides you with a variety of output based on the operands you specify.

Examples

Let's look at a few examples of how you might use the ACCEPT command.

Accepting PTF SYSMODs: After you have applied the SYSMODs into the target zone, you can define to SMP/E that you want to install only the PTF SYSMODs into the distribution zone. You can do this by specifying the following commands:

```
SET      BDY(ZOSDLB1).
ACCEPT   PTFS.
```

By issuing these commands, you direct SMP/E to accept all eligible PTF SYSMODs into distribution zone ZOSDLB1.

Suppose you do not want to install all the PTF SYSMODs, but only a select few. You can do this by specifying the following commands:

```
SET      BDY(ZOSDLB1).
ACCEPT   SELECT(UZ00001,UZ00002).
```

When you issue these commands, only PTFs UZ00001 and UZ00002 are installed in distribution zone ZOSDLB1.

Accepting SYSMODs for Selected Products: There may be times when you want to update only certain products on your system with the SYSMODs contained on a service tape. Assume you want to install all PTFs for a particular product. This can be accomplished by specifying the following commands:

Accepting the SYSMOD

```
SET      BDY(ZOSDLB1).
ACCEPT   PTFS
         FORFMID(HOP0001).
```

or

```
SET      BDY(ZOSDLB1).
ACCEPT   FORFMID(HOP0001).
```

In both cases, SMP/E accepts all applicable PTFs for the product whose FMID is HOP0001 and that is located in distribution zone ZOSDLB1. Unless you specify otherwise, PTFS is the default SYSMOD type.

Accepting SYSMODs Having Prerequisites: When installing a SYSMOD, you might not always know if it has prerequisites, or if the prerequisites are available. (Sometimes a prerequisite SYSMOD might not be received, or it might be held because it is in error.) In cases such as this, you can direct SMP/E to check whether an equivalent (or superseding) SYSMOD is available, by specifying the GROUPEXTEND operand.

Assume you want to process all PTFs for a product on your system, and you want to ensure that all other required SYSMODs are also processed. You can do this by specifying the following commands:

```
SET      BDY(ZOSDLB1).
ACCEPT   PTFS
         FORFMID(HOP0001)
         GROUPEXTEND.
```

By issuing these commands, you direct SMP/E to accept all PTFs, along with any other required SYSMODs, to the product whose FMID is HOP0001 and is located in the ZOSDLB1 target zone. If SMP/E cannot find a required SYSMOD, it looks for and uses a SYSMOD that supersedes the required one.

Accepting SYSMODs Using the CHECK Operand: In the previous example, SMP/E was directed to automatically include all SYSMODs needed for the specified product. There may be times when you want to see which SYSMODs are included before you actually install them. You can do this with the CHECK operand by issuing the following commands:

```
SET      BDY(ZOSTGT1).
ACCEPT   PTFS
         FORFMID(HOP0001)
         GROUPEXTEND
         CHECK.
```

After these commands are processed, you can check the SYSMOD Status report to see which SYSMODs would have been installed if you had not specified the CHECK operand. If you are satisfied with the results of this trial run, you can issue the commands again, without the CHECK operand, to actually install the SYSMODs.

For a more complete description of all the ACCEPT command operands and other examples, see The ACCEPT Command in *SMP/E Commands*.

Reporting Output

When ACCEPT processing is complete, these reports will help you analyze the results:

- The **SYSMOD Status report** provides you with a summary of the processing that took place for each eligible SYSMOD, based on the operands you specified on the ACCEPT command. It shows you which SYSMODs were accepted, which were not accepted, and why.
- The **Element Summary report** provides you with a detailed look at each element affected by ACCEPT processing. It tells you in which libraries the elements were installed.
- The **Causer SYSMOD Summary report** provides you with a list of SYSMODs that caused other SYSMODs to fail, and describes the errors that must be fixed to successfully process the SYSMODs. This report can reduce the amount of work involved in figuring out which errors caused SYSMODs to fail.
- The **File Allocation report** provides you with a list of the data sets used for ACCEPT processing and supplies information about these data sets.

Additional reports may be produced depending on the work being done and the content of the SYSMODs. For more information about all the reports produced by the ACCEPT command (and samples of actual reports), see The ACCEPT Command in *SMP/E Commands*.

Summary

Let's summarize what we have learned about using the ACCEPT command to install a SYSMOD in the distribution (or backup) libraries. The ACCEPT command:

- Selects SYSMODs to install
- Checks that all other required SYSMODs have been (or are being) installed
- Based on SYSMODs, selects elements to install
- Directs SMP/E to call the system utilities to update the distribution libraries
- Records what is accepted:
  - Distribution zone: Creates SYSMOD entries and element entries.
  - Global zone: Deletes SYSMOD entries and MCS statements in SMPPTS. Any SMPTLIB data sets created during RECEIVE processing are also deleted. (This global zone processing is optional.)
- Reports the results of processing

Remember, once you have accepted a SYSMOD, it **cannot** be restored!

Displaying SMP/E Data

You can use SMP/E to provide helpful information for planning new installations, debugging problems, and other instances when you want to know the function and service level of your product software. There are several ways you can display data in the SMP/E database.

In this chapter, you will learn about the kinds of information that help you manage your system and the best method by which the information can be obtained.

- **Query dialogs:** The easiest and fastest way to obtain just the information you want
- **LIST command:** When you need an all-inclusive hardcopy listing of information about your system
- **REPORT commands:** To check and compare the zone contents and generate command output that can be used to update your system
- **SMP/E CSI application programming interface:** To write an application program to query the contents of your system's CSI data sets.

Using the Query Dialogs

The SMP/E dialogs provide you with an online method of system management, software inventory, data base inquiries, and guidance. For example, with the Query dialogs, you can look up information in the CSI data set. The Query dialogs are one of the easiest and most direct methods you can use to obtain the content and status of any SYSMOD that has been processed by SMP/E. You can use the Query dialogs to display an entry in either a specific zone (CSI query) or in all zones (cross-zone query).

You can use the SMP/E dialogs to view a SYSMOD, even if it has been compacted. Use the Query dialog (zone is GLOBAL, entry type MCS, entry name is the SYSMOD name) and you will be shown the complete SYSMOD in an edit session. You may save the SYSMOD in a different location from this session. If you are using SMPPTS spill data sets, there is another benefit of viewing the SYSMOD from the Query dialog, in that you do not have to know in which SMPPTS data set the SYSMOD is stored; SMP/E will find it for you.

To get to the Query dialogs, you select SMP/E (option 1) on the initial SMP/E dialog panel (CIDPGV2). Then, on the main menu for SMP/E options (GIM@PRIM), select Query (option 3). This takes you to the initial Query panel, shown in Figure 17. If you need assistance with using the Query dialogs, (or any of the SMP/E dialogs), help panels are available.

Let's assume you want to find out which SYSMODs have been applied to a particular target zone on your system. You can accomplish this task using the QUERY SELECTION MENU and selecting the CSI QUERY option (1), as shown in Figure 17.

```
 GIMQUPO                        QUERY SELECTION MENU
===> 1

   1  CSI QUERY           - Display SMPCSI entries
   2  CROSS-ZONE QUERY    - Display status of an entry in
                            all zones
   3  SOURCEID QUERY      - Display SOURCEIDs for specified zone


   D  DESCRIBE            - Overview of using QUERY

   T  TUTORIAL            - Information on using QUERY

To return to the SMP/E primary option menu, enter  END .


5694-A01 5655-G44(C) COPYRIGHT IBM CORP 1982, 2003
```

Figure 17. Query Selection Menu

When the CSI QUERY panel is displayed (see Figure 18 on page 31), you can indicate that you want SMP/E to check target zone ZOSTGT1 for all SYSMOD entries.

```
GIMQU1PO                        CSI QUERY
===>

Specify the zone, entry type, and name to be queried:

      ZONE NAME      ===> ZOSTGT1 Name of the zone to be queried.
                                     To display a list of all zones,
                                     leave blank

      ENTRY TYPE     ===> SYSMOD  Entry type to be queried.
                                     To display a list of all valid
                                     entry types, leave ENTRY TYPE
                                     and ENTRY NAME blank

      ENTRY NAME     ===>         Entry name to be queried.
                                     To display a list of all entries
                                     for the selected zone and entry
                                     type, leave blank


To return to the Query selection menu, enter  END .
```

Figure 18. CSI Query Panel

Because the ENTRY NAME was left blank on the CSI QUERY panel, SMP/E displays another panel (see Figure 19) that lists all the SYSMOD entries in target zone ZOSTGT1.

```
GIMQUSEA               CSI QUERY - SELECT ENTRY
===>                                                  SCROLL ===> PAGE

 Select one entry to query from target zone ZOSTGT1 :

 S   NAME         ACTION
     AZ00005
 s   UZ00001
     UZ00002
```

Figure 19. CSI Query - Select Entry Panel

The CSI QUERY - SELECT ENTRY panel shows that SYSMODs AZ00005, UZ00001, and UZ00002 have been applied to target zone ZOSTGT1. If you want more information about the contents of SYSMOD UZ00001, you can select that entry by entering an **S** next to it, and another panel is displayed (see Figure 20).

```
GIMQIT26               CSI QUERY - SYSMOD ENTRY
===>                                                  SCROLL ===> PAGE

 To return to the previous panel, enter  END .


 Entry Type:  SYSMOD                              Zone Name:  ZOSTGT1
 Entry Name:  UZ00001                             Zone Type:  TARGET

   Type:  PTF                     Status:  APPLIED
   FMID:  HOP0001
   Date/Time:  02.341    13.57:31    APPLIED



   -------- -------- -------- -------- -------- -------- --------
 MODS    MOD01   MOD02
```

Figure 20. CSI Query - SYSMOD Entry Panel

The CSI QUERY - SYSMOD ENTRY panel displays all the relevant information pertaining to SYSMOD UZ00001.

As you can see, the QUERY dialog panels provide a quick and easy way for you to obtain information about your system.

Using the LIST Command

In the course of managing your system, there may be times when you need a hardcopy listing of some type of information. You can use the LIST command to accomplish this task. For example, it might be necessary for you to have a record of the following:

- All entries of a specific type
- Selected entries of a specific type
- All entries that meet certain criteria

The LIST command can provide you with a listing of this information.

Examples

Let's look at a few basic examples of how you might use the LIST command.

Listing Entries in a Particular Zone: In the course of managing your system, you might need to know which SYSMOD entries exist in the global zone. You can find this out by specifying the following commands:

```
SET     BDY(GLOBAL).
LIST    SYSMODS.
```

By issuing these commands, you direct SMP/E to list all the SYSMOD entries in the global zone.

Listing Specific Entries: Suppose you discover a problem on your system and need to determine whether a particular SYSMOD has been installed in the target zone. You can accomplish this by specifying the following commands:

```
SET     BDY(ZOSTGT1).
LIST    SYSMOD(UZ00001).
```

By issuing these commands, you direct SMP/E to provide you with information about SYSMOD UZ00001 in target zone ZOSTGT1.

Listing SYSMODs That Are Received but Not Installed: You might have received service into the global zone and are in the process of installing the service on your system. You want to see which of the SYSMODs you have received have not yet been installed in a target zone. This can be accomplished by specifying the following commands:

```
SET     BDY(GLOBAL).
LIST    SYSMODS
        NOAPPLY(ZOSTGT1).
```

By issuing these commands, you direct SMP/E to list the SYSMODs that have been received, but have not yet been applied to target zone ZOSTGT1.

Reporting Output

When LIST processing is complete, these reports will provide you with the information that was requested:

- The **LIST Summary report** provides you with information about the type of entry, name of entry, and status of entry for zones and data sets you have specified.

- The **File Allocation report** provides you with a list of the data sets used for LIST processing, and supplies information about these data sets.

For a more complete description of the LIST command, additional examples, and samples of actual reports, see The LIST Command in *SMP/E Commands*.

Using the REPORT Commands

You can use the REPORT commands to check and compare the SYSMODs installed in the different zones that exist on your system. In addition to this checking, you can tell SMP/E to generate the necessary commands to synchronize the specified zones. You can later modify these commands, if necessary, and use them to install the indicated SYSMODs.

One of the REPORT commands (REPORT SYSMODS) is very useful if you want to compare the SYSMODs installed in two zones. This command allows you to compare the following:
- One distribution zone to another distribution zone
- One target zone to another target zone
- A distribution zone to a target zone
- A target zone to a distribution zone

Example

Let's look at a basic example of how you might use the REPORT SYSMODS command. Assume you have two systems using the same global zone, and you want to check which SYSMODs are installed in a target zone on one system, but are not installed in a target zone on the other system. You can accomplish this by specifying the following commands:

```
SET    BDY(GLOBAL).
REPORT SYSMODS
       INZONE(ZOSTGT1)
       COMPARED(ZOSTGT2).
```

By issuing these commands, you direct SMP/E to compare the SYSMOD content of zone ZOSTGT1 to that of zone ZOSTGT2. Any SYSMODs that are in zone ZOSTGT1 and are not in zone ZOSTGT2 appear in the resulting report.

SMP/E also provides output you can use to install those SYSMODs you deem appropriate.

Reporting Output

When REPORT SYSMODs processing is complete, these reports will provide you with the information that was requested:
- The **SYSMOD Comparison report** provides you with a summary of the SYSMODs found in the input zone, but not found in the comparison zone. It can help you determine which SYSMODs might need to be installed in the comparison zone so its content reflects that of the input zone.
- The **File Allocation report** provides you with a list of the data sets used for REPORT processing, and supplies information about these data sets.

For a more complete description of the REPORT commands, additional examples, and samples of actual reports, see SMP/E Reports in *SMP/E Commands*.

SMP/E CSI Application Programming Interface

The SMP/E CSI application program interface (GIMAPI) allows you to write application programs that have read-only access to data stored in SMP/E's CSI (Consolidated Software Inventory) data sets. GIMAPI is described in detail in *SMP/E Reference*.

Summary

Let's summarize what you have learned about using the Query dialogs, the LIST command, the REPORT command, and the CSI API to check SMP/E's records for your system:

- Query dialogs: Easy and fast way to obtain information
- LIST command: Best for hardcopy listing
- REPORT commands: Best for checking and comparing zone contents
- SMP/E CSI application programming interface: Best for writing an application program to query the contents of your system's CSI data sets.

Chapter 2. SMP/E Concepts

This chapter summarizes some basic concepts that you will need to understand before you can use SMP/E. It briefly describes:
- What SMP/E is
- What system modifications are
- The data sets used by SMP/E
- How SMP/E can help you install and maintain products, and monitor changes to products

What Is SMP/E?

SMP/E is the basic tool for installing and maintaining software in OS/390 or z/OS systems and subsystems. It controls these changes at the element level by:
- Selecting the proper levels of elements to be installed from a large number of potential changes
- Calling system utility programs to install the changes
- Keeping records of the installed changes

SMP/E is an integral part of the installation, service, and maintenance processes for CBPDOs, ProductPacs, RefreshPacs, and selective follow-on service for CustomPacs. In addition, SMP/E can be used to install and service any software that is packaged in SMP/E system modification (SYSMOD) format.

SMP/E can be run either using batch jobs or using dialogs under Interactive System Productivity Facility/Program Development Facility (ISPF/PDF). SMP/E dialogs help you interactively query the SMP/E database, as well as create and submit jobs to process SMP/E commands.

These are some of the types of software that can be installed by SMP/E:
- Products and service provided in CBPDOs and CustomPac offerings
- Products and service from IBM Software Distribution Centers not provided in CBPDOs or CustomPac offerings
- Service provided in Expanded Service Options (ESOs)
- Other products and service

SMP/E can install software from any of these sources, provided it is packaged as a system modification, or *SYSMOD*.

What Are SYSMODs?

Software, whether it is a product or service, consists of *elements* such as macros, modules, source, and other types of data (such as CLISTs or sample procedures). For software to be installed by SMP/E, it must include control information for the elements. This information describes the elements and any relationships the software has with other products or service that may also be installed on the same system. The combination of elements and control information is called a system modification, or *SYSMOD*.

There are four types of SYSMODs:

- **Function SYSMODs (or functions).** These introduce a new product, a new version or release of a product, or updated functions for an existing product into the system.

  There are two types of function SYSMODs:

  – A *base function* either adds or replaces an entire functional area in the system. Examples of base functions are SMP/E and MVS.

  – A *dependent function* provides an addition to an existing functional area in the system. It is called dependent because its installation depends on a base function already being installed. Examples of dependent functions are the language features for SMP/E.
- **PTFs.** These are IBM-supplied, tested fixes for reported problems. They are meant to be installed in all environments. PTFs may be used as preventive service to avoid certain known problems that may have not yet appeared on your system, or they may be used as corrective service to fix problems you have already encountered. The installation of a PTF must always be preceded by that of a function SYSMOD, and often other PTFs as well.
- **APAR fixes.** Authorized program analysis reports (APARs) are temporary fixes designed to fix or bypass a problem for the first reporter of the problem. These fixes may not be applicable to your environment. The installation of an APAR must always be preceded by that of a function SYSMOD, and sometimes of a particular PTF. That is, an APAR is designed to be installed on a particular preventive-service level of an element.
- **User modifications (USERMODs).** These are SYSMODs built by you, either to change IBM code or to add independent functions to the system. The installation of a USERMOD must always be preceded by that of a function SYSMOD, sometimes certain PTFs, APAR fixes, or other USERMODs.

  Note: If you want to package a user application program or new system function in SMP/E format, the correct way is to build a base or dependent function SYSMOD, not a USERMOD.

Figure 21 on page 37 shows the hierarchy of the various SYSMOD types. This example shows two service chains: one for the base function HZY1101 and one for the dependent function JZY1121.

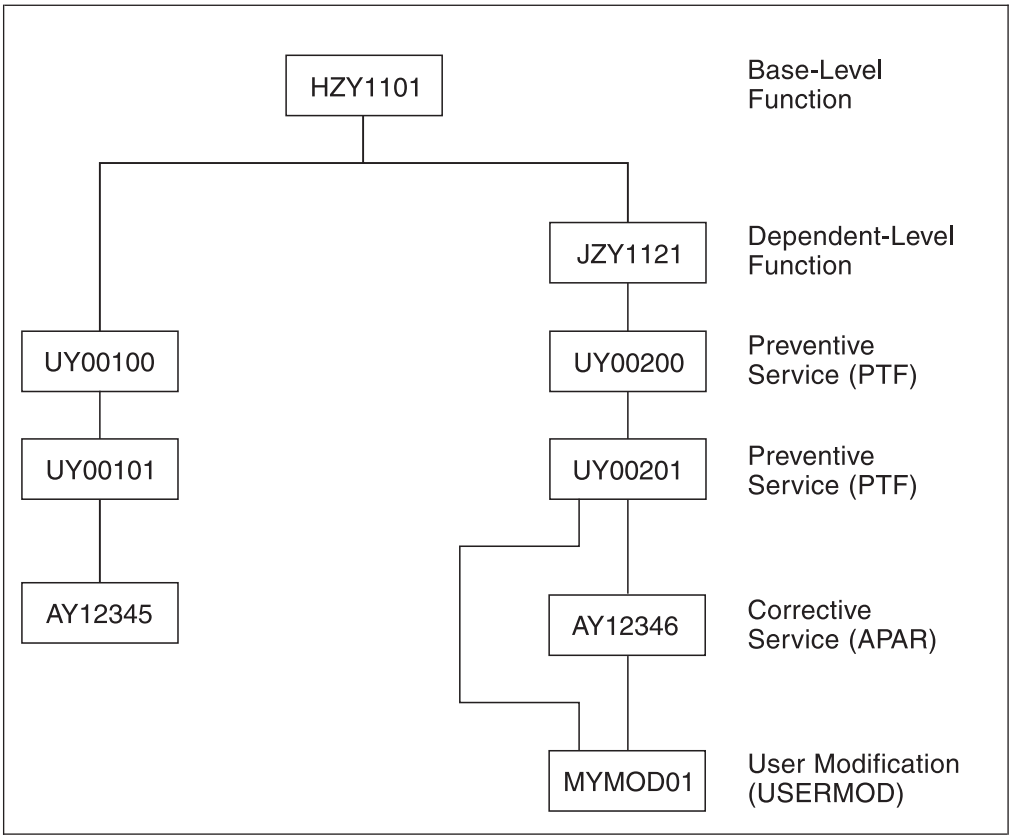


Figure 21. Example of a SYSMOD Hierarchy

SMP/E keeps track of the functional and service levels of each element and uses the SYSMOD hierarchy to determine such things as which functional and service levels of an element should be installed and the correct order for installing updates for elements. For more information about how SMP/E determines the processing order of changes, as well as the functional and service levels of elements, see The APPLY Command and The ACCEPT Command in *SMP/E Commands*.

Data Sets Used by SMP/E

When SMP/E processes SYSMODs, it installs the elements in the appropriate libraries and updates its own records of the processing it has done. SMP/E installs program elements into two types of libraries:

- **Target libraries** contain the executable code needed to run your system (for example, the libraries from which you run your production system or your test system).
- **Distribution libraries** (DLIBs) contain the master copy of each element for a system. They are used as input to the SMP/E GENERATE command or the system generation process to build target libraries for a new system. They are also used by SMP/E for backup when elements in the target libraries have to be replaced or updated.

To install elements in these libraries, SMP/E uses a database made up of several types of data sets:

- **SMPCSI (CSI) data sets** are Virtual Sequential Access Method (VSAM) data sets used to control the installation process and record the results of processing.

The CSI data set is a VSAM data set in which SMP/E maintains information about the system. A CSI can be divided into multiple partitions through the VSAM key structure. Each partition is referred to as a *zone*.

There are three types of zones:

– A single **global zone** is used to record information about SYSMODs that have been received into the SMPPTS data set. The global zone also contains information enabling SMP/E to access the other two types of zones, information about system utilities that SMP/E calls to install elements from SYSMODs, and information allowing you to tailor SMP/E processing.
– One or more **target zones** are used to record information about the status and structure of the operating system (or target) libraries. Each target zone also points to the related distribution zone, which can be used during APPLY, RESTORE, and LINK when SMP/E is processing a SYSMOD and needs to check the level of the elements in the distribution libraries.
– One or more **distribution zones** are used to record information about the status and structure of the distribution libraries (DLIBs). Each DLIB zone also points to the related target zone, which is used when SMP/E is accepting a SYSMOD and needs to check if the SYSMOD has already been applied.

Figure 22 shows the relationships between SMP/E zones and libraries.

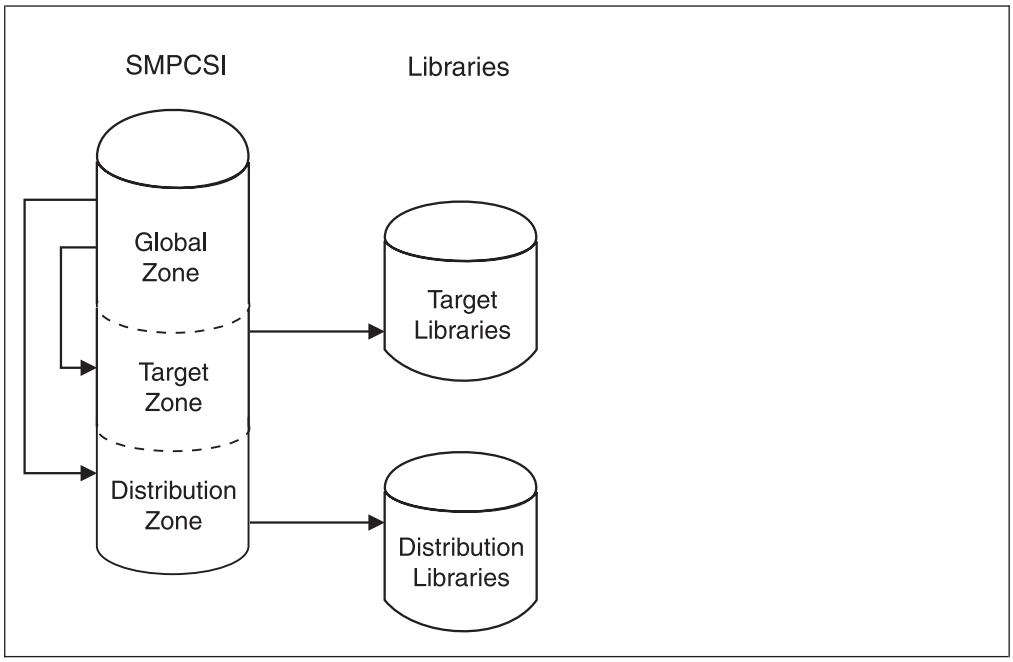


Figure 22. Summary of Zone Relationships

There can be more than one zone in an SMPCSI data set (in fact, there can be up to 32766 zones per data set). For example, an SMPCSI data set can contain a global zone, several target zones, and several distribution zones. The zones can also be in separate SMPCSI data sets. One SMPCSI data set can contain just the global zone, a second SMPCSI data set the target zones, and a third SMPCSI data set the distribution zones. For more information on ways to structure SMPCSI data sets, see the *SMP/E Reference* manual.

- An **SMPPTS (PTS) data set** is a data set for temporary storage of SYSMODs waiting to be installed.

The PTS is used strictly as a storage data set for SYSMODs. The RECEIVE command stores SYSMODs directly on the PTS without any modifications of SMP/E information. The PTS is related to the global zone in that both data sets contain information about the received SYSMODs. Only one PTS can be used for a given global zone. Therefore, you can look at the global zone and the PTS as a pair of data sets that must be processed (for example, deleted, saved, or modified) concurrently.

- The **SMPSCDS (SCDS) data set** contains backup copies of target zone entries modified during APPLY processing. Therefore, each SCDS is directly related to a specific target zone, and each target zone must have its own SCDS.

  SCDS data sets are used by SMP/E to store backup copies of target zone entries modified during APPLY processing. Therefore, each SCDS is directly related to a specific target zone, and each target zone must have its own SCDS.

SMP/E also uses the following data sets:

- The **SMPMTS (MTS) data set** is a library in which SMP/E stores copies of macros during installation when no other target macro library is identified. Therefore, the MTS is related to a specific target zone, and each target zone must have its own MTS data set.
- The **SMPSTS (STS) data set** is a library in which SMP/E stores copies of source during installation when no other target source library is identified. Therefore, the STS is related to a specific target zone, and each target zone must have its own STS data set.
- The **SMPLTS (LTS) data set** is a library that maintains the base version of a load module. The load module in this library specifies a SYSLIB allocation in order to implicitly include modules. Therefore, the LTS is related to a specific target zone, and each target zone must have its own LTS data set.
- **Other utility and work data sets**.

SMP/E uses information in the CSI data sets to select proper element levels for installation, to determine which libraries should contain which elements, and to identify which system utilities should be called for the installation.

System programmers can also use the CSI data sets to obtain the latest information on the structure, content, and status of the system. SMP/E provides this information in reports, listings, and dialogs to help you:

- Investigate function and service levels
- Understand intersections and relationships of SYSMODs (either installed or waiting to be installed)
- Build job streams for SMP/E processing

How SMP/E Can Help You Install and Maintain Products

This section briefly describes the tasks involved in general SMP/E processing, installing SYSMODs, and monitoring and maintaining your system, as well as the commands used to accomplish these tasks. Following is a guide to the commands that help you with these tasks. (You can use SMP/E dialogs for all the tasks in this list.) For more information about these commands, see the *SMP/E Commands* manual.

Where to Begin

You must specify a SET command before SMP/E can process any other commands.

Specifying the Zone to Be Processed: SET

The main SMP/E database is a set of CSIs. When you establish the SMP/E database, you use the UCLIN command or the administration dialogs to divide the CSI into one or more partitions called *zones*. A zone contains control information for the following:

- A set of target libraries. (These zones are called *target zones*.)
- A set of distribution libraries. (These zones are called *distribution zones*.)
- The data present in the PTS and for general SMP/E processing. (This zone is called the *global zone*.)

You must tell SMP/E which zone you are working with before it can execute any other commands. You direct SMP/E processing to a specific zone by coding the zone name on the SET command.

Installing SYSMODs

The primary purpose of SMP/E is to install SYSMODs. This section describes the tasks and commands you can use.

Loading SYSMODs into the SMPPTS: RECEIVE

The RECEIVE command performs the following functions:

- Reads in SYSMODs from a sequential input file, defined by the SMPPTFIN DD statement
- Reads in HOLDDATA for exception SYSMODs from another sequential input file, defined by the SMPHOLD DD statement
- Determines which SYSMODs and HOLDDATA are applicable to your system
- Stores the SYSMODs and HOLDDATA in a storage data set (the PTS data set) and in the global zone

Note: The RECEIVE command can also be used to transfer software packages from a TCP/IP network connected server directly into an SMP/E directory or data sets.

Installing SYSMODs in the Target Libraries: APPLY

After the SYSMODs have been received, you can use the APPLY command to direct SMP/E to install them in the appropriate target libraries.

Installing SYSMODs in the Distribution Libraries: ACCEPT

After installing a SYSMOD in the target libraries and testing it to your satisfaction, you can use the ACCEPT command to have SMP/E install that SYSMOD in the distribution libraries and delete it from the PTS and the global zone.

Note: Though the usual order of processing a SYSMOD is RECEIVE, APPLY, and then ACCEPT, you skip the APPLY step and go directly from RECEIVE to ACCEPT. For further information, see Chapter 4, “Installing a New Function”, on page 83.

Building a System from a Set of Distribution Libraries: GENERATE

Use the GENERATE command to create a job stream that builds a system (a set of target libraries) from a set of distribution libraries.

Monitoring Your System

This section describes the tasks and commands you can use to monitor your system.

Displaying Information from the SMP/E Database: LIST

Use the Query dialogs or the LIST command to display the data SMP/E retains. There are several operands you can use to list subsets of the data.

Checking and Comparing Zone Contents: REPORT

The REPORT command helps you obtain information about SYSMODs installed on your system. There are several types of REPORT commands that you can use to do the following:

- **REPORT CROSSZONE** to list conditional requisites that must be installed in certain zones because of SYSMODs installed in other zones. This information can help you synchronize service for related products that are in different zones.
- **REPORT ERRSYSMODS** to list SYSMODs that have already been installed but are affected by error holds subsequently received.
- **REPORT SOURCEID** to list source IDs assigned to SYSMODs in a given zone or ZONESET.
- **REPORT SYSMODS** to list SYSMODs installed in one zone and applicable to a second zone, but not yet installed in it.

Managing the SMP/E Database

This section describes the tasks and commands you can use to remove SYSMODs and process the SMP/E database as a part of your system maintenance.

Notes:

1. There is no SMP/E command to remove a SYSMOD from the distribution libraries once you have accepted it.
2. Because all SMP/E processing is controlled by information in the CSI data set, you must provide the required information in the CSI before you process any SYSMODs.

Removing SYSMODs From the Target Libraries: RESTORE

After you have installed a SYSMOD in the target libraries, you should test it to make sure the change works correctly. If during testing you find an error in one of the SYSMODs you applied, you can use the RESTORE command to remove that SYSMOD from the system.

The RESTORE command causes the elements in the distribution libraries to be reinstalled on the target libraries. Therefore, when restoring a SYSMOD, you may have to restore more than the one SYSMOD in error. All SYSMODs that have not been accepted and that affect some of the same elements or need the same service level requisites as the one you are restoring must also be restored.

Removing SYSMODs From the PTS: REJECT

After receiving a SYSMOD, you can use the REJECT command to remove that SYSMOD from the PTS and the global zone. This can be done to rework a SYSMOD.

You can also use REJECT after you use NOPURGE to accept a SYSMOD into the distribution libraries. Using NOPURGE in the OPTIONS entry prevents SMP/E from deleting the global zone SYSMOD entry and the PTS MCS entry during ACCEPT processing. Later, you can delete the SYSMOD and MCS entries by using REJECT.

Processing the SMP/E Database to Create, Update, or Delete a Single Entry: UCLIN

You can use the Administration dialogs or the UCLIN command to create or change entries in CSI data sets. UCLIN can be used to create entries required during SMP/E processing, such as entries for:

- Identifying the utilities SMP/E is to use
- Providing information for dynamic allocation support

You can also use UCLIN to correct errors in entries or to alter processing information.

Note: This command should be used with **extreme** caution; an incorrect change can cause a SYSMOD to be installed incorrectly later.

UCLIN updates only entries in SMP/E data sets. It does nothing to any elements or load modules in any product libraries. You must ensure that the appropriate changes are made to the libraries.

Processing the SMP/E Database to Update Several Entries of the Same Type: ZONEEDIT

Use the ZONEEDIT command to quickly change the values for a field in different DDDEF or UTILITY entries in the same zone. You can also use ZONEEDIT to change the cross-zone subentries of MOD, LMOD, and TARGETZONE entries.

Processing the SMP/E Database to Define the Structure of the Target Libraries: JCLIN

To install a SYSMOD successfully, SMP/E must have information about the structure of your target libraries, such as:

- The library in which an element resides
- How modules are link-edited together to form load modules
- Where the load modules exist and their characteristics

The JCLIN command enables you to define the target library structure. In some instances, such as defining the target library structure for data elements, it is not necessary to use JCLIN, because the definition in the MCS statement is sufficient.

When processing the JCLIN command, you provide SMP/E with a job stream containing all the job steps (such as copies, link-edits, and assemblies) needed to create a set of target libraries from a set of distribution libraries. SMP/E then scans that input and builds all required entries to define the target system structure.

Processing the SMP/E Database to Write Information to the SMPLOG Data Set: LOG

Use the LOG command to write to the SMPLOG and SMPLOGA data sets. This is useful for maintaining documentation about your system, such as who installed a certain SYSMOD, and why.

Processing the SMP/E Database to Clean Up the SMPLTS, SMPMTS, SMPSTS, and SMPSCDS Data Sets: CLEANUP

Use the CLEANUP command to delete entries from the SMPMTS, SMPSTS, and SMPSCDS data sets for a particular target zone, if any entries still exist after the associated SYSMOD has been accepted into the related distribution zone.The CLEANUP command can also be used to remove the base version of load modules that are no longer needed from the SMPLTS data set.

Managing Zones

This section describes the tasks and commands you can use to manage zones.

Copying a Zone from One CSI to Another: ZONECOPY

Use the ZONECOPY command to copy an entire target or distribution zone from one CSI data set to another. When you use the ZONECOPY command, SMP/E checks that the zone you copy into is empty except for a ZPOOL record.

Copying a Zone to a Sequential Data Set: ZONEEXPORT

The ZONEEXPORT command creates a copy of a specified distribution, target, or global zone and places it into another data set (a variable-block sequential data set). Having this copy of a specified zone gives you two advantages:

- You can use it as a backup copy to recreate a zone that has been accidentally erased or destroyed.
- You can use it as a "transportable" copy to create the same zone on another system or in another CSI data set on the same system.

Copying a Zone from a Sequential Data Set to a CSI Data Set: ZONEIMPORT

Use the ZONEIMPORT command to reload an exported zone into a distribution, target, or global zone. ZONEIMPORT can be used only with output from ZONEEXPORT.

Deleting a Zone: ZONEDELETE

Sometimes you need to delete the SMP/E data related to one of the systems you are supporting. These are some examples of when you need to delete a zone:

- After a full system generation, you have to delete the information describing the previous target system libraries. Then you rebuild that information to describe the new set of target system libraries built from the distribution libraries.
- After installing a new level of a product that existed in its own target zone and distribution zone, you want to delete the information about the old level of the product and continue processing only the new level.

Merging Zones: ZONEMERGE

Use the ZONEMERGE command to merge one zone into another zone after you use the GENERATE command or do a full system generation. When you use the ZONEMERGE command, SMP/E looks through the zones that ZONEMERGE works on and merges the data from them.

Renaming a Zone: ZONERENAME

Use the ZONERENAME command to rename a zone. This command can help you keep meaningful names for your zones. You can also use ZONERENAME after a GENERATE command or a full system generation to help you change the name of the distribution zone that GENERATE or the system generation uses to build a new target zone.

Linking and Relinking Modules

This section describes the tasks and commands you can use to link or relink modules into load modules.

Handling Cross-Zone Link-Edits: LINK MODULE

Use the LINK MODULE command to link modules in one zone with load modules in another zone and to track this inclusion so that subsequent APPLY and RESTORE processing can automatically maintain the affected load modules.

Relinking Load Modules That Use CALLLIBS: LINK LMODS

Use the LINK LMODS command to relink load modules that have a CALLLIBS subentry list in their LMOD entry (and therefore use the automatic library call option to implicitly include modules from a specified library). The LINK LMODS command may also be used to relink specific load modules that do not contain CALLLIBS, rebuilding them from scratch if necessary.

General SMP/E Processing

This section describes general SMP/E processing tasks and the commands you can use to do them.

Requesting Diagnostic Processing: DEBUG

Use the DEBUG command to display the name of the issuing routine in each SMP/E message, to dump SMP/E control blocks, to dump VSAM request parameter list (RPL) control blocks, and to get a SNAP dump of SMP/E and its work areas whenever a specified message is issued.

Resetting Return Codes: RESETRC

Normally, the execution of an SMP/E command depends on the successful execution of all preceding commands. However, you can use the RESETRC command to reset the return code to zero. This allows SMP/E to run the next command, even if a preceding command failed.

Chapter 3. Preparing to Use SMP/E

This chapter discusses how to prepare to use SMP/E after it has been installed. It describes how to do the following:

1. Allocate and initialize data sets in the SMP/E database.
2. Define entries in the CSI data set to do the following:
   - Create the zones associated with the PTS, distribution libraries, and target libraries.
   - Define the product and SMP/E libraries used during SMP/E processing.
   - Define the utility programs and associated parameters used during SMP/E processing.
   - Define the libraries that are eligible for retry processing after *x*37 abends.
3. Connect the SMP/E dialogs to ISPF
4. Set up SMP/E for easier operation:
   - Specify SMP/E OPTIONS entry
   - Specify link edit utility output DDDEF entries
   - Specify automatic cross-zone requisite checking
5. Define the information needed to invoke SMP/E.
6. Define exit routines, if desired, to customize SMP/E processing.

Allocating and Initializing Data Sets in the SMP/E Database

To install SYSMODs, SMP/E needs information about them and about the target and distribution libraries where they are to be installed. This information is kept in a database composed of the following data sets:

- SMPCSI (CSI)
- SMPPTS (PTS)
- SMPSCDS (SCDS)

CSI Data Sets

SMP/E uses CSIs to keep records of the system. To define the CSI data sets for your system, you need to do the following:

1. Decide how to organize the CSI data sets.
2. Understand catalog considerations for CSI data sets.
3. Allocate the CSI data sets.
4. Define alias entries for user catalogs.
5. Initialize the CSI data sets.
6. Define the zones for your system.
7. Understand how to reorganize a CSI data set to reclaim space.

Deciding How to Organize CSI Data Sets

Before you allocate any CSI data sets, you must decide how to organize those data sets. Consider the following when you make your decision:

- The DASD configuration of your system libraries
- The organization of your system support structure
- How you want to use SMP/E

There are two basic structures for CSI data sets:

- Single-CSI
- Multiple-CSI

Descriptions of these structures are followed by examples.

Single-CSI Structure: You can define the CSI structure to have one CSI that keeps track of all your system activity. The single-CSI data set has one global zone and one or more target and distribution zones. These are some reasons for having a single-CSI data set:

- The single-CSI data set optimizes the use of direct access storage.
- The single-CSI data set puts your whole establishment in one VSAM data set. This provides you with a single control point and one source of information for your whole system.

Single-CSI systems do have a drawback. When SMP/E needs to process a zone, it cannot request access to that specific zone; it must request access to the CSI data set containing that zone. As a result, if you have a single-CSI system, you can run only one background SMP/E job at a time.

Figure 23 shows a single-CSI data set structure.

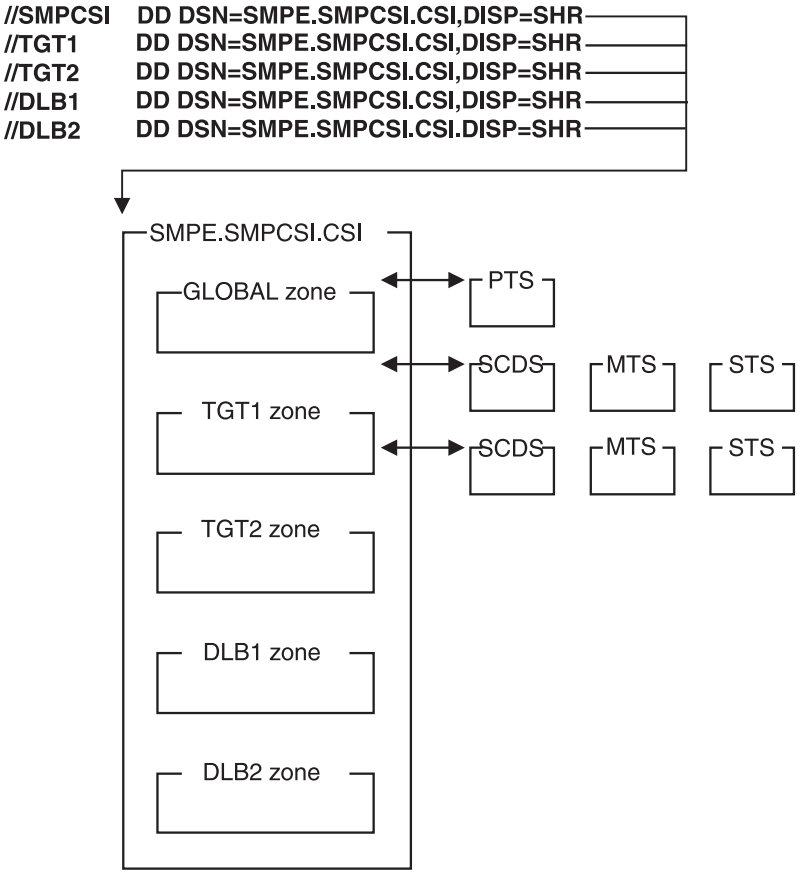


Figure 23. Sample Single-CSI Structure

Multiple-CSI Structure: Multiple CSI data sets can be:

- Separate from each other, each with its own global zone.
- Connected by ZONEINDEX entries to a single global zone. (The global zone must be in one of the CSI data sets.)

Multiple CSIs enable you to use more than one VSAM data set for the global, target, and distribution zones. These are some reasons for having multiple CSI data sets:

- Your system may need multiple CSIs because of the characteristics of a particular installation—its programming support, its backup and update needs, and its need for added security and data integrity. For example, keeping libraries and their associated zones synchronized when you dump them for backup is easier if you keep them on the same physical DASD.
- Your system may need multiple CSIs if the support teams for different subsystems—such as MVS, CICS, IMS, and NCP—are at different places.
- You may want to be able to run more than one background SMP/E job at a time. When SMP/E needs to process a zone, it cannot request access to that specific zone; it must request access to the CSI data set containing that zone. If your zones are in separate CSI data sets, processing for one zone does not prevent access to another zone.

Figure 24 shows a multiple-CSI data set structure.

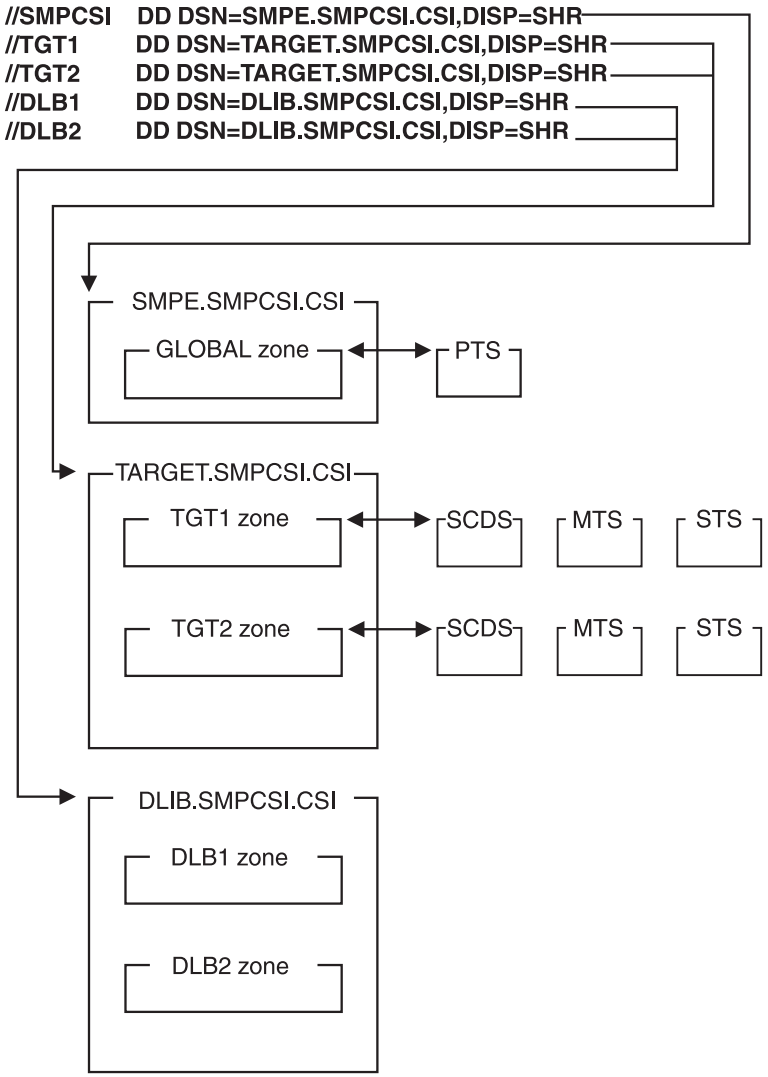


Figure 24. Sample Multiple-CSI Structure

Examples of CSI Structures: The following examples show several ways to define a CSI structure describing a sample z/OS system with Job Entry Subsystem 3 (JES3) and an Information Management System (IMS) subsystem. In some of the examples, a single CSI contains multiple zones. Others show zones that are separated into multiple CSI data sets. Zones in separate CSIs can be connected by a single global zone. The CSI that contains the global zone is called the *master CSI*.

Example 1: Using a Separate Global Zone for Each Subsystem: In Figure 25, the existing DASD structure for the libraries is maintained, and a separate global zone is defined for each of the subsystems (MVS, JES3, and IMS). This CSI structure keeps control of the three subsystems separate. You might use this structure if the system programming support for the three subsystems is organizationally separate. A disadvantage is that there is no single control point (global zone) from which to manage or query the entire system.

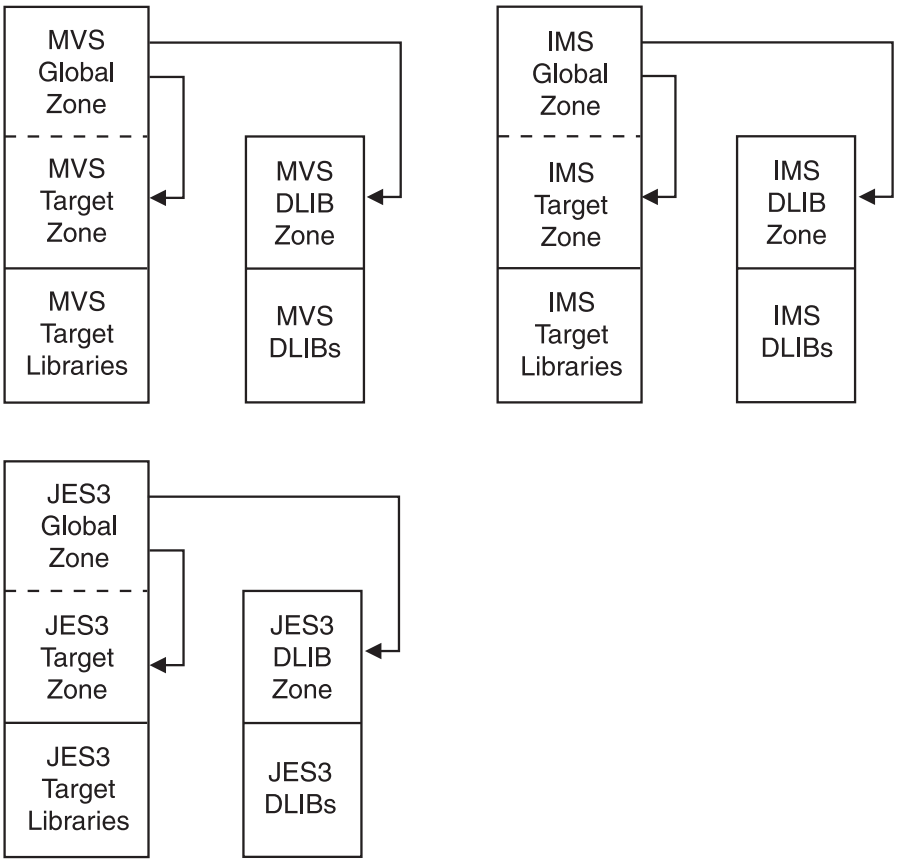


Figure 25. Using a Separate Global Zone for Each Subsystem

Example 2: Using a Single CSI for the Whole System: In Figure 26 on page 49, all the SMP/E control information for the system is contained in a single CSI. The system structure is reflected by separate zones for MVS, JES3, and IMS. This CSI structure provides a single control point from which to manage or query the entire system.

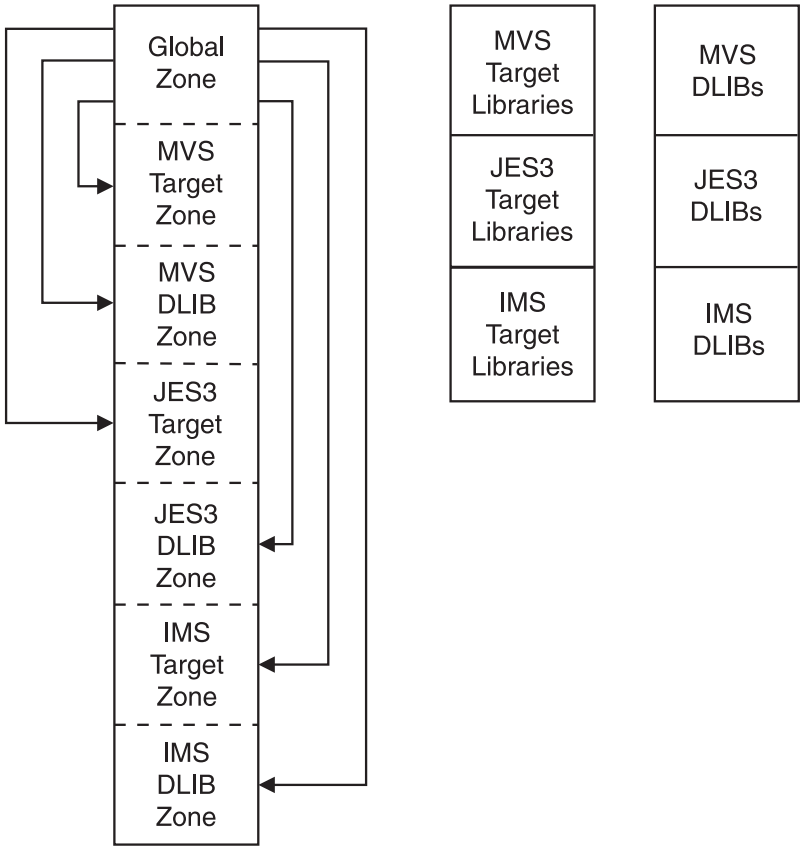


Figure 26. Using One CSI for the Whole System

Example 3: Using a Master CSI and Multiple CSIs: In Figure 27 on page 50, one global zone is defined for the entire system in a master CSI, and separate CSIs are allocated for the JES3 and IMS subsystems on the packs where the subsystem libraries reside. This CSI structure provides the advantages of a common control point (as in Example 2) and keeps the SMP/E control information physically associated with the libraries it describes. This is useful when you dump packs for backup.

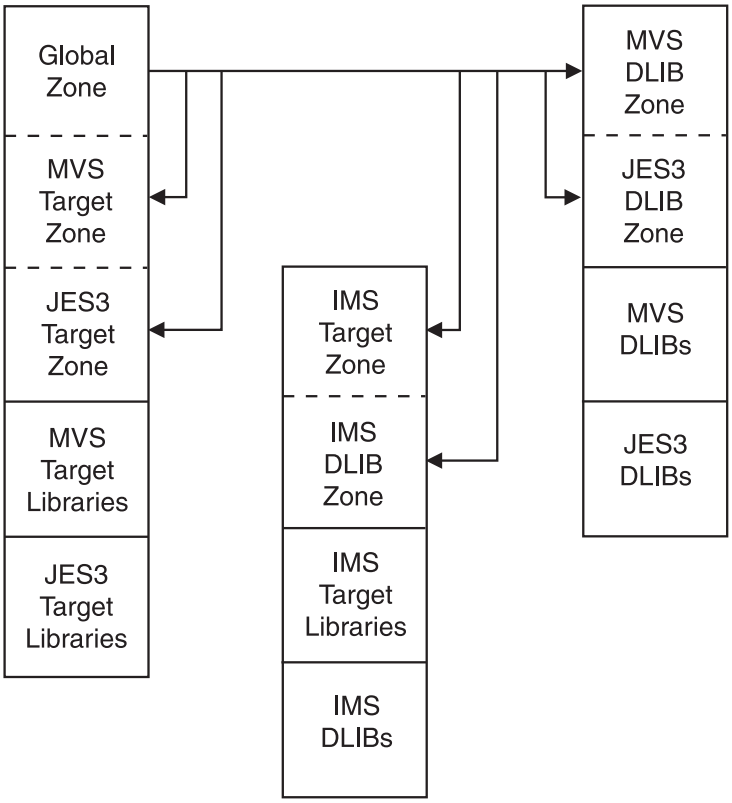


Figure 27. Using a Master CSI

Example 4: Using a Master CSI and a Separate CSI for Each Zone: In Figure 28 on page 51, one global zone is defined for the entire system in a master CSI, and separate CSIs are allocated for the JES3 and IMS subsystems on the packs where the subsystem libraries reside. Unlike Example 3, each zone is in its own separate CSI. This CSI structure provides the advantages of a common control point (as in Example 2) and keeps the SMP/E control information physically associated with the libraries it describes. This is useful when you dump packs for backup.

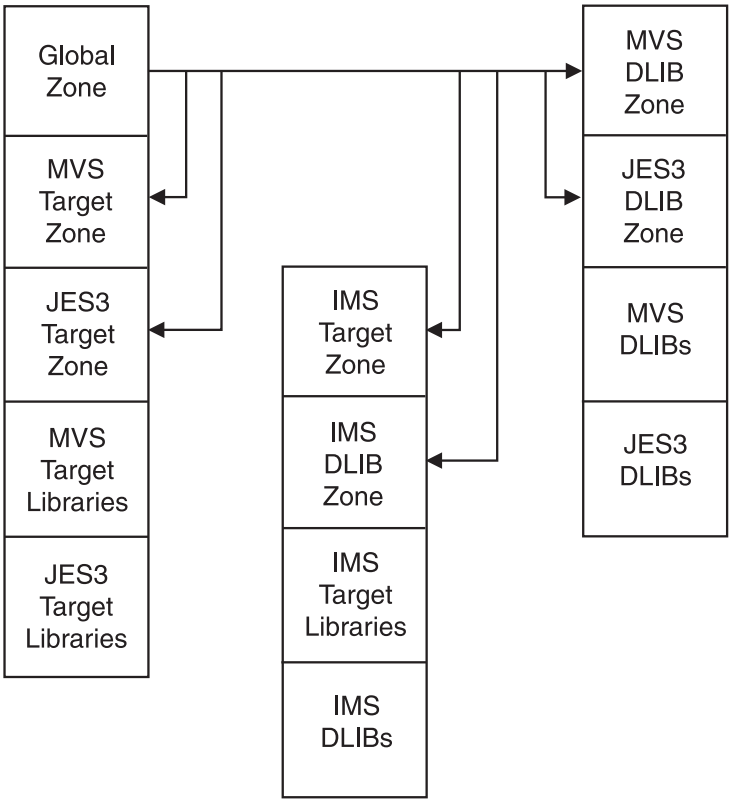


Figure 28. Using a Master CSI and a Separate CSI for Each Zone

Example 5: Using a Master CSI and One CSI per SREL: In Figure 29 on page 52, one global zone is defined for the entire system in a master CSI, and separate CSIs are allocated for each SREL (MVS, CICS, NCP, and IMS/DB2). The target zones and DLIB zones associated with a given SREL are in the same CSI. This CSI structure provides the advantages of a common control point through one global zone (as in Example 2) and keeps the SMP/E control information physically associated with the libraries it describes. This is useful when you dump packs for backup.

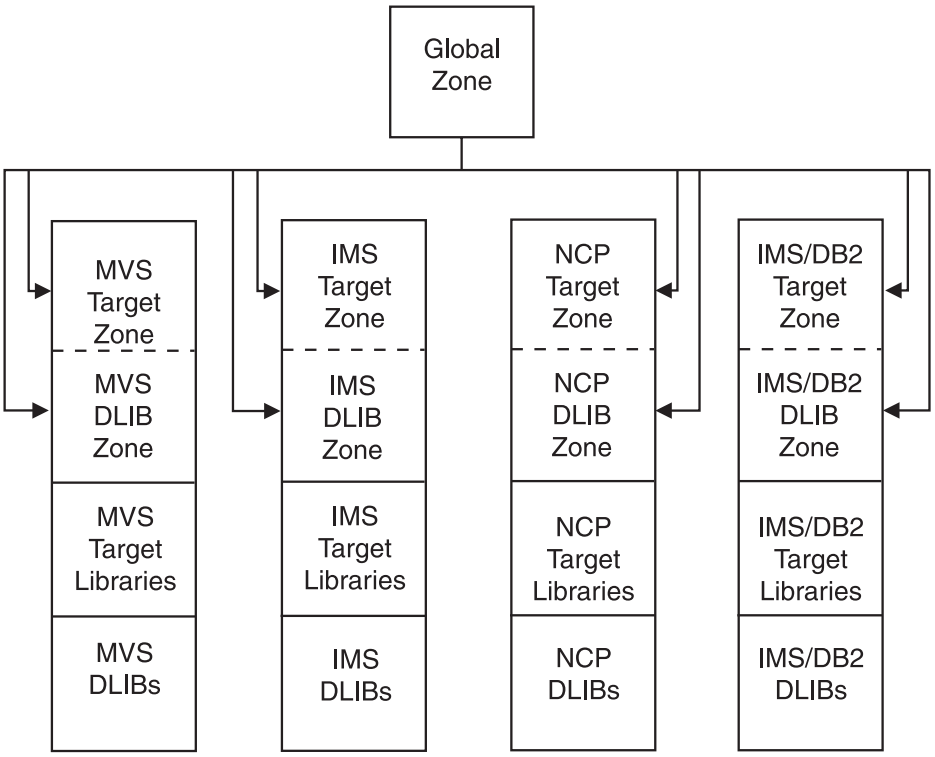


Figure 29. Using a Master CSI and One CSI per SREL

Catalog Considerations

When you catalog the CSI data sets used by SMP/E, remember these considerations:

- **User catalogs:** You should catalog each CSI in a user catalog, not in the master catalog. However, the user catalog does not need to be on the same volume as the CSI.
- **Alias entries for user catalogs:** Catalog information should be accessible through your master catalog. One way to do this is to define each user catalog as an alias in the master catalog. For an example of defining an alias for a user catalog, see "Defining an Alias Entry for a User Catalog" on page 54. Defining alias entries for user catalogs enables you to access all the CSI data sets without having to use a STEPCAT DD statement. Also, it eliminates the need to restore both the DASD containing the master catalog and the DASD containing the CSI after an I/O error.

  If no alias is defined for the user catalog, you must include a STEPCAT DD statement each time SMP/E is called.

Allocating a CSI Data Set

CSI data sets are keyed VSAM clusters and are allocated by use of access method services. For additional information and a description of the parameters, see *z/OS DFSMS Access Method Services for Catalogs*.

The GIMSAMPU member in SYS1.SAMPLIB is a sample job to allocate and prime CSI and SMP/E operational data sets. The following sample job step, which is taken from the sample job in GIMSAMPU, allocates a CSI data set with enough space to have multiple target or distribution zones and then initializes the CSI with the zpool record:

```
//DEFZONES EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//GIMZPOOL DD DSN=SYS1.MACLIB(GIMZPOOL),DISP=SHR
//SYSIN    DD *
   DEFINE CLUSTER(                          +
            NAME(SMPE.GLOBAL.CSI)           +
            VOLUMES(volid)            +
            CYLINDERS(100 10)               +
            FREESPACE(10 5)                 +
            KEYS(24 0)                      +
            RECORDSIZE(24 143)              +
            SHAREOPTIONS(2 3)               +
                )                           +
       DATA (                               +
            NAME(SMPE.GLOBAL.CSI.DATA)      +
            CONTROLINTERVALSIZE(8192)       +
            )                               +
      INDEX (NAME(SMPE.GLOBAL.CSI.INDEX)    +
            CONTROLINTERVALSIZE(4096)       +
            )
   REPRO INFILE(GIMZPOOL)                   +
            OUTDATASET(SMPE.GLOBAL.CSI)
 /*
```

In your own job, be sure to include:

- NAME

  These are the naming conventions for CSI data sets:

  – The high-level qualifier must not be **SYS1** if the CSI data set is cataloged in a user catalog rather than in the master catalog. However, the user catalog should be accessible through an alias entry in the master catalog.

  – The low-level qualifier must be **CSI.**

  These are examples of SMP/E data set names:

  ```
  'SMPE.SMPCSI.CSI'
  'PP.SMPCSI.CSI'
  'IMS.SMPCSI.CSI'
  'TEST.CSI'
  ```

- KEYS(24 0)
- RECORDSIZE(24 143)
- SHAREOPTIONS(2 3)

  SMP/E requires **2** as the cross-region SHAREOPTIONS value. It uses the default value of 3 as the cross-system SHAREOPTIONS value. Because SMP/E does not support cross-system sharing of the CSI, you cannot specify **4** as the cross-system value for SHAREOPTIONS. If you want to support cross-zone sharing, you must either use Global Resource Serialization (GRS) or a similar function, or ensure that the data set is not updated by multiple systems simultaneously.

- CONTROLINTERVALSIZE (CISIZE)

  If you allocate more than one CSI, give each CSI the same control interval (CI) size.

  Note: If you are operating in a multiple CSI environment, make sure of the following:
  – That the index CI sizes are equal.
  – That the data CI sizes are equal.

- CYLINDERS

The CYLINDERS value is only an **estimated** starting value. Your cylinder value may vary according to the device type, the software arrangement, the amount of service you install, and the number of CSIs.

Defining an Alias Entry for a User Catalog

After allocating the CSI data sets, you should define alias entries for the high-level qualifiers of your CSI data sets in your master catalog and relate them to your SMP/E user catalog. This lets you access the new CSI data sets without having to use a STEPCAT DD statement.

The following job creates an alias entry in the master catalog for a CSI data set named **SMPE.SMPCSI.CSI** that is cataloged in a user catalog named **SMPECAT**:

```
//CREATE   JOB    'accounting info',MSGLEVEL=(1,1)
//ALIAS    EXEC   PGM=IDCAMS
//SYSPRINT DD     SYSOUT=A
//SYSIN    DD     *
  DEFINE ALIAS                      -
         (NAME(SMPE)                -
         RELATE(SMPECAT))           -
         CATALOG(AMASTCAT/password)
/*
```

If the CSI data sets are cataloged in different user catalogs, they must have different high-level qualifiers.

Defining Zones for Your System

Once you have allocated and initialized the CSI data sets, you need to create within them the entries SMP/E uses to maintain your system. The first entries you need to define are the *zone definition entries* — GLOBALZONE, TARGETZONE, and DLIBZONE entries —,which set up zones in CSI data sets.

- **GLOBALZONE entry.** A global zone is created by defining a GLOBALZONE entry. The GLOBALZONE entry contains processing-related information for SMP/E. It is also used by SMP/E as an index to target and distribution zones, either in the same CSI or in different CSI data sets. The GLOBALZONE entry must be defined before you can do any other processing for that global zone.
- **TARGETZONE entry.** A target zone is created by defining a TARGETZONE entry. The TARGETZONE entry contains information SMP/E uses to process a specific target zone and the associated target libraries. It must be defined before you can do any other processing for that target zone.
- **DLIBZONE entry.** A distribution zone is created by defining a DLIBZONE entry. The DLIBZONE entry contains the information SMP/E uses to process a specific distribution zone and the associated distribution libraries. It must be defined before you can do any other processing for that distribution zone.

Figure 30 on page 55 illustrates how zone definition entries define the relationships between zones.

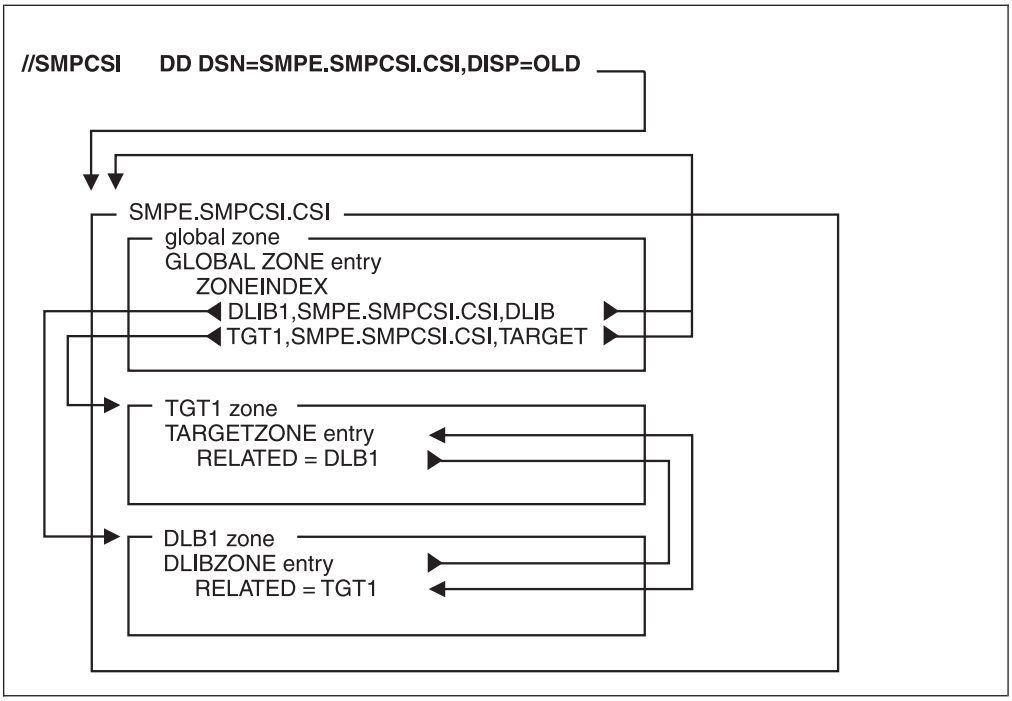


Figure 30. Relationships between Zone Definition Entries

After you have defined the zones for your system, you can create other entries. SMP/E zones contain two basic types of entries:

- Entries controlling SMP/E processing.

  You generally define processing control entries through the SMP/E Administration dialogs or with the UCLIN command. Table 2 summarizes the information specified in these entries.
- Entries describing the structure and status of the target and distribution libraries.

  Status and structure entries are generally created by SMP/E when you install SYSMODs, run the JCLIN command, or copy entries from one zone to another. Table 3 on page 56 summarizes the information specified in these entries.

SMP/E provides a member in SYS1.SAMPLIB (GIMSAMPU) containing sample UCLIN statements to define entries for a basic z/OS system. You can access this member by use of standard system utilities. The sample definitions are syntactically correct and can be used as the basis for your CSI entries. This sample is not complete for all systems, but it is an example of the types of information various entries need. For examples of UCLIN to define entries, see The UCLIN Command in *SMP/E Commands*, which shows the UCLIN syntax for each entry type, and SMP/E Data Set Entries in *SMP/E Reference*, which contains a description of the syntax plus examples and notes on its use.

Table 2. Entries Controlling SMP/E Processing

Type of Information	Entry Type	Zone Where Defined		
		Global	Target	DLIB
Data set definitions for dynamic allocation[1]	DDDEF	X	X	X
DLIB zone processing information	DLIBZONE			X
Exception (held) SYSMODs[2]	HOLDDATA[2]	X		
FMIDs to limit the SYSMODs processed by an SMP/E command	FMIDSET	X		

Table 2. Entries Controlling SMP/E Processing (continued)

Type of Information	Entry Type	Zone Where Defined		
		Global	Target	DLIB
Global zone processing information	GLOBALZONE	X		
Processing options[3]	OPTIONS	X		
Target zone processing information	TARGETZONE		X	
Utility program parameters[4]	UTILITY	X		
Zone names to limit the SYSMODs processed by an SMP/E command	ZONESET	X		

Notes:
1. For more information about dynamically allocating data sets, see "How to Dynamically Allocate Data Sets to Be Used During SMP/E Processing" on page 58.
2. For more information about processing exception SYSMODs, see Chapter 8, "Managing Exception SYSMODs". HOLDDATA entries cannot be updated by UCLIN or the Administration dialogs.
3. For more information about defining information to be used during SMP/E's retry processing after *x*37 abends, see "Recovering After Errors from Utility Processing" on page 64.
4. For more information about defining utility programs and associated parameters, see "Defining Utility Programs and Associated Parameters to SMP/E" on page 60.

Table 3. Entries Describing the Status and Structure of the Target and Distribution Libraries

Type of Information	Entry Type	Zone Where Defined		
		Global	Target	DLIB
Assembler statements that can be assembled to create an object module	ASSEM		X	X
Data elements installed in the target or distribution libraries (data elements are elements other than macros, modules, or source)	Data element entries		X	X
Distribution libraries that were totally copied to target libraries	DLIB		X	X
Elements installed in a hierarchical file system (HFS)	Hierarchical file system element entries		X	X
Java Archive files	JAR		X	X
Java Archive update files	JARUPD		X	X
Load module information	LMOD		X	X
Macros that have been installed in the target or distribution libraries	MAC		X	X
Module source that has been installed in the target or distribution libraries	SRC		X	X
Modules used to create load modules in the target libraries	MOD		X	X
Program objects	PROGRAM		X	X
Software product information	PRODUCT	X		
Software product feature information	FEATURE	X		
SYSMODs that have been processed	SYSMOD	X	X	X

Reorganizing a CSI Data Set to Reclaim Space

During normal SMP/E processing, VSAM control interval splits and control area splits can occur. These splits use up some of the free space in each control interval or control area, and this can degrade SMP/E performance and DASD space utilization. You should monitor your VSAM data sets regularly to determine how many splits have occurred and how much free space remains. The following job lists the catalog entry for data set **SMPE.SMPCSI.CSI**:

```
//LISTCAT  JOB    'accounting info',MSGLEVEL=(1,1)
//STEP01   EXEC   PGM=IDCAMS
//SYSPRINT DD     SYSOUT=A
//SYSIN    DD     *
  LISTCAT                          -
         ENTRIES(SMPE.SMPCSI.CSI) -
         ALL
/*
```

After examining the LISTCAT output, you may determine that the CSI should be reorganized to eliminate splits in the control intervals or control areas and to reset the amount of free space available. This can be done through the access method services EXPORT and IMPORT commands. Once a CSI has been exported, a new CSI can be allocated, and the exported CSI can be imported so that normal SMP/E processing can continue.

Note: These examples are not the only way of compressing the CSI. You may prefer to use another method, drawing on your experience and knowledge of VSAM.

The following is a sample job for exporting the CSI:

```
//EXPORT   JOB    'accounting info',MSGLEVEL=(1,1)
//STEP01   EXEC   PGM=IDCAMS
//SMPCSI   DD     DSN=SMPE.SMPCSI.CSI,DISP=OLD
//TEMPCSI  DD     DSN=SMPCSI.DATA,DISP=OLD
//SYSPRINT DD     SYSOUT=A
//SYSIN    DD     *
  EXPORT SMPE.SMPCSI.CSI  -
         INFILE(SMPCSI)   -
         OUTFILE(TEMPCSI)
/*
```

The following is an sample job for importing the CSI:

```
//IMPORT   JOB    'accounting info',MSGLEVEL=(1,1)
//STEP01   EXEC   PGM=IDCAMS
//SMPCSI   DD     DSN=SMPE.SMPCSI.CSI,DISP=OLD
//TEMPCSI  DD     DSN=SMPCSI.DATA,DISP=OLD
//SYSPRINT DD     SYSOUT=A
//SYSIN    DD     *
  IMPORT  INFILE(TEMPCSI)  -
          OUTFILE(SMPCSI)  -
          INTOEMPTY
/*
```

Notes:

1. If you want to delete the original CSI (SMPE.SMPCSI.CSI) when the exported copy (SMPCSI.DATA) is created, do **not** use the IDCAMS TEMPORARY keyword on the EXPORT command.

   If you want to make a backup copy of the CSI, you can use the TEMPORARY keyword on the EXPORT command to keep the original CSI intact.
2. Use a sequential data set to receive the exported CSI.

3. After allocating a new CSI to be imported into, do **not** prime it with the GIMZPOOL record provided in SYS1.MACLIB; if you do, the import operation will fail.

PTS Data Sets

The PTS data set is used as temporary storage for SYSMODs. It contains one member for each SYSMOD received. Each member is called a modification control statement (MCS) entry and is an exact copy of the SYSMOD as it was received from the SMPPTFIN data set. The name of an MCS entry matches the ID of the SYSMOD it contains. Generally, the MCS entries are kept on the PTS until the SYSMOD is accepted; then, under normal processing, they are deleted.

Each PTS data set must be associated with only one global zone. The allocation of space and directory blocks for the SMPPTS depends on your plans for installing and maintaining the functions managed by the global zone. For more information about allocating the SMPPTS data set, see *SMP/E Reference*.

SCDS Data Sets

The SCDS data set contains backup copies of target-zone entries that are modified during APPLY processing. These backup copies are made before the entries are (1) changed by inline JCLIN, a ++MOVE MCS, or a ++RENAME MCS or (2) deleted by an element MCS with the DELETE operand. The backup copies are used during RESTORE processing to return the entries to the way they were before APPLY processing.

Each backup copy of an entry is associated with the SYSMOD that caused the entry to be backed up. Together, the collection of entries associated with a SYSMOD is called the *BACKUP* entry for that SYSMOD. When you process the SCDS (for example, to list entries), you can specify only BACKUP entries; you cannot process individual entries within a BACKUP entry.

Each target zone within a CSI must have its own SCDS. The correct SCDS to be used during processing is determined either by the SMPSCDS DDDEF entry in each specific target zone or by a DD statement in the JCL. An SCDS can be allocated the same way as any normal partitioned data set. The allocation of space and directory blocks for this data set depends on your plans for installing and maintaining functions. For more information about allocating the SMPSCDS data set, see *SMP/E Reference*.

How to Dynamically Allocate Data Sets to Be Used During SMP/E Processing

The processing of SMP/E commands requires a variety of data sets. You can either provide the DD statements for these data sets (such as in a cataloged procedure) or have SMP/E allocate the data sets dynamically. Dynamic allocation has the advantage that data sets are allocated only as they are needed; DD statements must successfully allocate all data sets, regardless of whether they are needed for the command being processed.

There are some drawbacks to using DD statements. For example, all the data sets defined by DD statements must be successfully allocated, regardless of whether they are needed for the command being processed. In addition, if you are running several SMP/E commands, you must be careful to use the correct DD statements

for each command. If you are processing zones that are in different CSI data sets, you must make sure to provide a DD statement that points to each of those zones and their associated CSIs.

With dynamic allocation, you do not have these problems. Subsequent sections describe the sources from which SMP/E can get the information it needs to allocate data sets dynamically and how it chooses which of these sources to use.

Sources of Information for Dynamic Allocation

SMP/E can use several sources of information to allocate data sets dynamically:
- DDDEF entries
- SMPPARM member GIMDDALC
- Standard defaults

DDDEF Entries

You can use DDDEF entries to provide SMP/E with information it needs to allocate any of the following:
- Permanent data sets, such as target libraries, distribution libraries, and SMP/E data sets
- Temporary data sets
- SYSOUT data sets
- Work data sets
- Pathnames for elements and load modules residing in a hierarchical file system (HFS)

The name of the DDDEF entry must match the ddname of the data set it describes and the entry must exist in the zone that uses the data set. DDDEF entries provide more flexibility than DD statements; they enable different zones to use different data sets for the same ddname and they use resources more efficiently because they allow SMP/E to allocate only the data sets it needs.

DDDEF entries can include the following information:
- Data set name
- Unit type
- Volume serial number
- Initial data set status: NEW, OLD, MOD, or SHR
- Final data set status: KEEP, DELETE, or CATALOG
- How the data set is to be allocated: blocks, cylinders, or tracks
- Primary and secondary values for space allocation
- Whether the data set should be RACF-protected
- Whether the data set is SMS-managed
- Directory information used to allocate the pathname for an element or load module residing in a hierarchical file system (HFS).

For more information about DDDEF entries, see the *SMP/E Reference* manual.

SMPPARM member GIMDDALC

Another way to provide SMP/E with information about data sets is through GIMDDALC control statements in SMPPARM member GIMDDALC. For more information about GIMDDALC, see the chapter on SMPPARM members in the *SMP/E Reference* manual.

Standard Defaults: SMPOUT and SYSPRINT are critical for SMP/E to operate properly. Therefore, in case they are not defined, SMP/E has built-in defaults for them.

- SMPOUT is allocated either as SYSOUT (for background processing) or to the terminal (for foreground processing).
- SYSPRINT is allocated as SYSOUT.

How Dynamic Allocation Works

Once SMP/E has determined which data sets are needed for the command it is processing, SMP/E checks whether DD statements have been provided for any of those data sets. SMP/E uses information from those DD statements in allocating the data sets to which they apply. If any data sets lack DD statements, SMP/E must allocate them dynamically. To get the information it needs to do this, SMP/E checks the following sources in the order shown.

1. **DDDEF entries.** If the zone specified on the SET command contains a DDDEF entry for the required data set, SMP/E uses that entry to allocate the data set. Otherwise, it checks the next source.
2. **SMPPARM member GIMDDALC.** If a GIMDDALC control statement has been defined in SMPPARM member GIMDDALC, SMP/E uses that information to allocate the data set. Otherwise, it checks the next source.
3. **Standard defaults.** If the data set is for SMPOUT or SYSPRINT, SMP/E uses the standard default to allocate the data set. Otherwise, data set allocation fails.

Notes:

1. When a data set is part of a concatenation (such as the SYSLIB concatenation), SMP/E does not do the previously described checking. All data sets in a concatenation must be defined the same way. For example, if a DDDEF entry specifies a concatenation, all the specified entries must also be defined by DDDEF entries.
2. For the cross-zone phase of APPLY and RESTORE processing, a DD statement cannot be used to allocate the SYSLIB for a cross-zone load module. This library can be allocated only through a DDDEF entry in the cross-zone containing the LMOD entry for the cross-zone load module.
3. The DD name **SMPDUMMY** is always allocated as a DUMMY data set. SMP/E ignores any allocation information specified for SMPDUMMY by a DDDEF entry or GIMDDALC member. If SMPDUMMY was previously allocated outside of SMP/E, SMP/E frees the SMPDUMMY DD and then reallocates it as a DUMMY data set.

Defining Utility Programs and Associated Parameters to SMP/E

SMP/E calls utility programs to install SYSMODs in the target and distribution libraries. These utilities must reside in either the LNKLST concatenation or the link pack area (LPA), as defined in SYS1.PARMLIB. SMP/E uses default utility programs unless you define OPTIONS entries and UTILITY entries specifying other utilities and parameters.

- OPTIONS entries point to the specific UTILITY entry to be used for each type of utility. These are the types of utilities you can point to:
  - Access methods services
  - Assembler
  - Compress
  - Copy
  - Link-edit
  - Retry after *x*37 abends
  - Update
  - Superzap

- Each UTILITY entry defines the information to be used when invoking a specific type of utility:
  - The name of the utility program
  - The maximum utility return code that SMP/E should consider to be successful
  - The ddname to be used for utility output
  - Parameters to be passed to the utility

Using Default Values for Utility Programs

If you do not define UTILITY entries and OPTIONS entries to specify which utility programs to use, SMP/E uses default utility programs, as well as its own default values, for return codes, print values, and the parameters to be passed. Table 4 lists the default values for the various types of utility programs.

Table 4. Default Values for UTILITY Entries

Utility	NAME	RC	PRINT	PARM
Access method services	IDCAMS	0	SYSPRINT	
Assembler	ASMA90	4	SYSPRINT	XREF, NOOBJECT, DECK
Compress	IEBCOPY	0	SYSPRINT	
Copy	IEBCOPY	0	SYSPRINT	
Hierarchical file system copy	BPXCOPY	0	SYSPRINT (see note 3)	
Link-edit utility	IEWBLINK or IEWL (see note 1)	8	SYSPRINT	LET, LIST, NCAL, XREF (see note 2)
Retry after *x37* abends	IEBCOPY	0	SYSPRINT	
Update	IEBUPDTE	0	SYSPRINT	Determined by SMP/E during processing
Superzap	IMASPZAP	4	SYSPRINT	

Notes:

1. When the DFP level supports the binder, IEWBLINK is the default link-edit utility. When the DFP level does not support the binder, IEWL is the default.
2. When the load module being link-edited contains a CALLLIBS subentry list, SMP/E does not always use NCAL by default. In this case, SMP/E uses CALL for the link to the actual target library or NCAL for the link to the SMPLTS library. SMP/E always uses NCAL for ACCEPT processing.
3. If SYSTSPRT is specified as the PRINT value, it is ignored and the default of SYSPRINT is used instead.

Defining Values for Utility Programs

If you want to use utility programs other than the defaults, or if you want to specify different parameters for the default utility programs, you need to identify the programs to SMP/E by defining UTILITY entries and OPTIONS entries. For example, the installation information for a particular product you are about to

install may direct you to use a specific link-edit utility and may indicate that the maximum successful return code from the utility is 4, not 8.

Figure 31 shows how OPTIONS entries, UTILITY entries, zone definition entries, and the SET command are related.

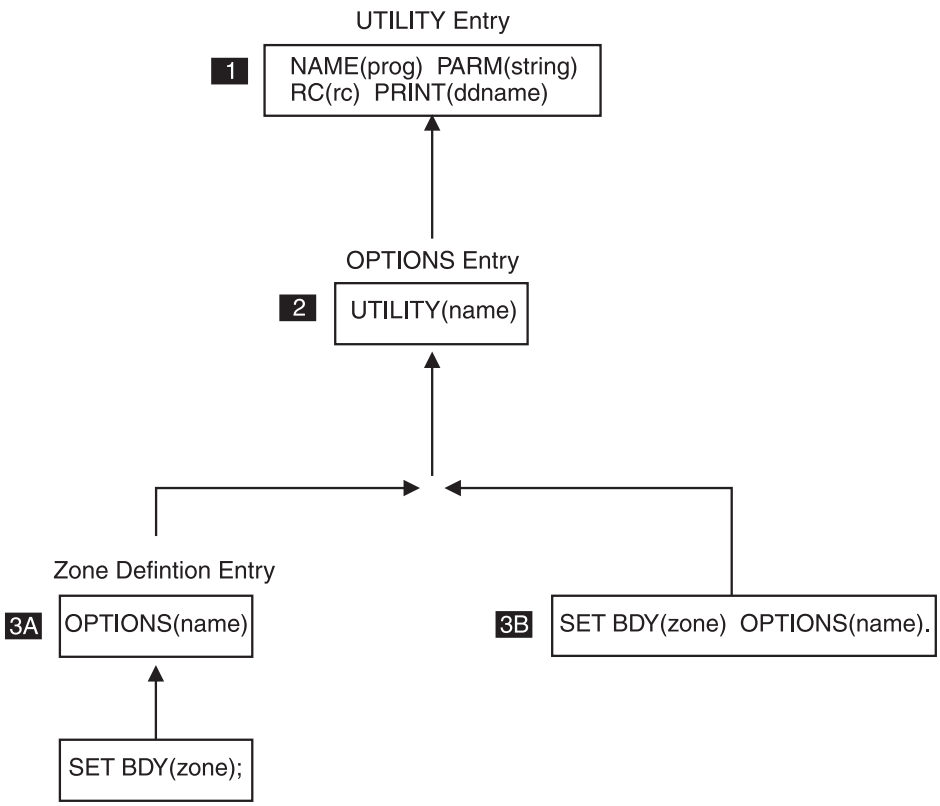


Figure 31. Relationships of OPTIONS, UTILITY, Zone Definition Entries and the SET Command

These are the basic steps you follow:

1 Define the desired utility name and parameters in a UTILITY entry.

2 Define an OPTIONS entry that points to that UTILITY entry.

3 Put the OPTIONS entry into effect by doing one of the following:

- **A** If the information should be the default for processing a particular zone, update the associated zone definition entry to point to the desired OPTIONS entry. The default OPTIONS ENTRY is always used for processing that zone, unless you override the OPTIONS entry with the SET command.
- **B** Otherwise, use the SET command to indicate which OPTIONS entry to use when processing the zone specified on the SET command. The information in the specified OPTIONS entry overrides the default OPTIONS entry defined for the zone.

For examples of these steps, see "Example: How to Request the Desired Utility Processing" on page 63. For more detailed information, see OPTIONS Entry, UTILITY Entry, GLOBALZONE Entry, DLIBZONE Entry, and TARGETZONE Entry in *SMP/E Reference*, and the SET command in *SMP/E Commands*.

Note: You can also restrict the utility programs that SMP/E can use by modifying the information in module GIMUTTBL, which is provided by SMP/E. For more information, see the GIMUTTBL chapter in *SMP/E Reference*.

Example: How to Request the Desired Utility Processing

Table 5 describes the steps you need to follow in order to get the desired utility processing. For details on syntax and coding considerations, see the GIMUTTBL chapter in *SMP/E Reference*.

Table 5. How to Request the Desired Utility Processing

Steps	Sample Scenario

1 Define the desired utility name and parameters in a UTILITY entry.	You want SMP/E to call a user routine, USERRCVR, rather than IEBCOPY, to recover from *x*37 abends. This is the processing you need:
	• The program must receive parameter **TYPE=FAST**.
	• The output should go to X37PRINT rather than SYSPRINT.
	• A return code of 4 or less is acceptable.
	• You want to suppress the listing of member names during retry processing done by your program.
	
	The following UCL defines the desired UTILITY entry for your program:
	`SET BDY(GLOBAL) /* Set to global zone. */.`
	`UCLIN /* */.`
	`ADD UTILITY(MYX37) /* Retry/recovery program. */`
	`NAME(USERRCVR) /* Program name. */`
	`PARM(TYPE=FAST) /* PARM value. */`
	`PRINT(X37PRINT) /* SYSPRINT ddname. */`
	`RC(4) /* Highest acceptable */`
	`/* return code. */`
	`LIST(NO) /* No list of member names. */`
	`/* */.`
	`ENDUCL /* */.`

2 Connect the UTILITY entry to an OPTIONS entry.	Once you have created the desired UTILITY entry, you need to point to it from an OPTIONS entry. The following UCL defines an OPTIONS entry (MYOPT1) that points to UTILITY entry MYX37:
	`SET BDY(GLOBAL) /* Set to global zone. */.`
	`UCLIN /* */.`
	`ADD OPTIONS(MYOPT1) /* New OPTIONS entry. */`
	`RETRY(MYX37) /* Connect to retry. */`
	`/* */.`
	`ENDUCL /* */.`

3A Use the zone definition entry to specify your OPTIONS entry as the default OPTIONS entry.	You might want your OPTIONS entry to be the default for processing target zone TGT1. In this case, the TARGETZONE entry for TGT1 must point to your OPTIONS entry. The following UCL updates the existing TARGETZONE entry for TGT1 so it points to OPTIONS entry MYOPT1:
	`SET BDY(TGT1) /* Set to target zone, TGT1.*/.`
	`UCLIN /* */.`
	`REP TARGETZONE /* Update zone definition. */`
	`OPTIONS(MYOPT1) /* OPTIONS entry to be used.*/`
	`/* */.`
	`ENDUCL /* */.`

Table 5. How to Request the Desired Utility Processing (continued)

Steps	Sample Scenario

3B Use the SET command to have your OPTIONS entry override the default OPTIONS entry for the zone.	Instead of changing the default OPTIONS entry, you might want to override whatever the default might be and use OPTIONS entry (MYOPT1) for processing particular commands or SYSMODs. In this case, the SET command preceding the commands you want to process must point to your OPTIONS entry. The following UCL points the SET command to OPTIONS entry MYOPT1:
	`SET BDY(TGT1) /* Set to target zone, TGT1.*/`
	`OPTIONS(MYOPT1) /* OPTIONS entry used. */.`
	⋮

Recovering After Errors from Utility Processing

To complete as many updates as possible to your product libraries, SMP/E tries to recover from errors that occur during processing by the utilities it invokes. The type of recovery SMP/E attempts for such errors depends on the type of error that occurred and the type of utility that was called.

- **Batched updates, no out-of-space problems.** Items to be processed by the link-edit utility or the copy utility are often batched. If an error other than an *x*37 abend occurs during such a utility call, SMP/E debatches the items and reinvokes the utility to attempt updates for individual members. This recovery is attempted automatically by SMP/E; no user intervention is required.
- **Out-of-space errors (*x*37 abends).** If a list of data sets is specified by the user, SMP/E attempts "retry" processing for data sets that have run out of space. During retry processing, the data set that ran out of space is compressed, and the utility is called again to retry the updates.

  If retry processing does not reclaim sufficient space and input to the utility was batched (copy or link-edit utility only), SMP/E debatches the input and retries the utility for each member separately. If this final attempt fails, the resulting *x*37 abend is treated as an unacceptable utility return code, and processing continues for other eligible updates.

This section explains what you need to define in order to have SMP/E attempt retry processing for *x*37 abends.

Overview of Your Input to Retry Processing

Your input to retry processing is through subentries in the OPTIONS entry, an optional retry exit routine, and the RETRY command operand.

- **OPTIONS entry.** The RETRYDDN and EXRTYDD subentries in the OPTIONS entry indicate which libraries are eligible for retry processing.

  The RETRYDDN subentry specifies which libraries should be included in retry processing. Without this list of libraries, SMP/E does not attempt retry processing.

  The EXRTYDD subentry specifies which libraries should be excluded from retry processing. This makes it easier for you to include all but a few specific libraries in retry processing.
- **Exit routine.** The retry exit routine enables you to control retry processing when an *x*37 abend occurs, instead of having SMP/E compress the out-of-space data set and reinvoke the failing utility.

If SMP/E determines that a retry can be attempted, it cancels the abend dump and calls the retry exit routine. The routine can then either cancel retry processing or perform some other method of recovery.

- **RETRY operand.** The RETRY operand tells SMP/E whether to attempt retry processing for the specific SMP/E command that is being processed. RETRY can be specified on the ACCEPT, APPLY, LINK LMODS, LINK MODULE, and RESTORE commands.

  You do not need to specify this operand in order to request retry processing, because the default is RETRY(YES). However, you can explicitly specify RETRY(YES) if you want to.

  To prevent retry processing for a specific command, specify RETRY(NO) instead of using RETRY(YES).

Example: How to Request the Desired Retry Processing

Table 6 describes the steps you need to follow in order to get the desired retry processing. For details on syntax and coding considerations, see *SMP/E Reference*.

Table 6. How to Request the Desired Retry Processing

Steps	Sample Scenario
1 Decide which data sets should be included in or excluded from retry processing.	You want retry processing for all libraries except LINKLIB, MIGLIB, and NUCLEUS.

2 Define the necessary subentries in the OPTIONS entries that you plan to use.	You already have been using OPTIONS entry OPT1 for all your ACCEPT, APPLY, LINK LMODS, LINK MODULE, and RESTORE commands. You need to add RETRYDDN and EXRTYDD values to specify the libraries that are eligible for retry processing.
	`SET BDY(GLOBAL).`
	`UCLIN .`
	`ADD OPTIONS(OPT1)`
	`RETRYDDN(ALL)`
	`EXRTYDD(LINKLIB,MIGLIB,`
	`NUCLEUS)`
	`.`
	`ENDUCL .`

3 Decide whether to use the default RETRY UTILITY entry, or to point to and define a RETRY UTILITY entry that specifies the desired information.	You will use the defaults for the retry utility as shown in Table 4 on page 61.
4 If desired, define an exit routine for retry processing. The Retry Exit Routine section in *SMP/E Reference* provides all the information you need about the interface for this routine and what SMP/E expects the routine to do.	You will use SMP/E's retry processing instead of writing a retry exit routine.

Table 6. How to Request the Desired Retry Processing (continued)

Steps	Sample Scenario

5 Make sure the desired OPTIONS entry is in effect for the zone you are processing.
 	
The methods you can use are shown in Figure 31 on page 62.
	As indicated in step 2, you have been using OPTIONS entry OPT1. Instead of specifying it on the SET command, you have defined TZONE entry (TGT1) and DZONE entry (DLIB1), which specify OPT1 as the OPTIONS entry to be used. (These zones are already defined in the global zone by ZONEINDEX subentries.)
	
	`SET BDY(TGT1).`
	`UCLIN .`
	`ADD TARGETZONE(TGT1)`
	`OPTIONS(OPT1)`
	`SREL(Z038)`
	`RELATED(DLIB1)`
	`.`
	`ENDUCL .`
	`SET BDY(DLIB1).`
	`UCLIN .`
	`ADD DLIBZONE(DLIB1)`
	`OPTIONS(OPT1)`
	`SREL(Z038)`
	`RELATED(TGT1)`
	`.`
	`ENDUCL .`

6 Use RETRY(YES) on the commands you want retry processing to be done for (RETRY can be specified on the ACCEPT, APPLY, LINK LMODS, LINK MODULE, and RESTORE commands.).
Note: To prevent retry processing for a specific command, specify RETRY(NO) instead.
	You are installing product HYY2102 and the related service, and you want SMP/E to attempt retry processing. You choose to use RETRY(YES) as the default instead of specifying it explicitly.
	
	`SET BDY(TGT1).`
	`APPLY FORFMID(HYY2102)`
	`FUNCTIONS PTFS`
	`GROUPEXTEND`
	`.`

Connecting SMP/E Dialogs to ISPF

The SMP/E dialogs run under ISPF. Therefore, if you plan to use the dialogs, you must connect them to ISPF. Follow these steps:

1. Make sure you have the required programs on your system.
2. If you have the TSO/VS2 programming control facility (PCF), add the necessary dialog modules to the PCF command table.
3. Concatenate the dialog libraries in a logon procedure or a command list (CLIST).
4. Connect the dialogs to ISPF.
5. Customize the dialogs, if desired.

These steps are described in the sections that follow.

Check for Required Programs

Make sure the following programs are on your system:

- ISPF (Version 4 Release 2, or later)
- ISPF/PDF (Version 2 Release 3, or later)
- TSO/E (Version 2 Release 5, or later)

The SMP/E dialogs use the TSO OUTPUT, SUBMIT, and STATUS commands. Therefore, to support these commands, you must provide a TSO IKJEFF53 user

installation exit routine that does not restrict the TSO OUTPUT and STATUS commands to job names beginning with the user's user ID. For details, see *z/OS TSO/E Customization*.

Add Dialog Modules to the PCF Command Table

If you have the TSO/VS2 programming control facility (PCF), add the following modules to the PCF command table:

GIMPSBTP
GIMISID1
GIMISID2
GIMISID3
GIMISID4
GIMISID5

Concatenate the Dialog Libraries

Table 7 lists the SMP/E dialog libraries needed to concatenate with ISPF libraries To use the dialogs, concatenate these libraries in a TSO logon procedure or in a CLIST. Here are some recommendations for concatenating the libraries:

- There are two sets of dialog libraries for SMP/E: one for English and one for Japanese. You can include the libraries for only **one** of these features in a given logon procedure or CLIST. If you want to be able to use both features on the same system, you need a logon procedure or CLIST for each feature. To switch from one feature to the other, you have to exit ISPF, run the procedure or CLIST for the other feature, and get back into ISPF.
- Do not use the ISPF LIBDEF service to concatenate the following libraries. Instead, use either the TSO ALLOCATE command or DD statements and JCL.
  - Dialog CLISTs (GIM.SGIMCLS0): If you use LIBDEF, the CLISTs and EXECs in the data set cannot be executed.
  - Dialog load module library (GIM.SGIMLMD0): If LIBDEF is used to concatenate the load module libraries, the SMP/E dialogs cannot find required load modules.

Table 7. ISPF Libraries and Related SMP/E Target Libraries

DDNAME	SMP/E Library	Contents
SYSPROC	GIM.SGIMCLS0	Dialog CLISTs
ISPLLIB	GIM.SGIMLMD0	Dialog load modules
ISPMLIB	GIM.SGIMMENU	Dialog messages
ISPPLIB	GIM.SGIMPENU	Dialog panels
ISPSLIB	GIM.SGIMSENU	Skeleton JCL procedures
ISPTLIB	GIM.SGIMTENU (see Note 1)	Table input library
ISPCTL1	(see Note 2)	Generated JCL
ISPCTL2	(see Note 2)	Generated JCL
SMPTABL	A user-defined name (see Note 3)	SMP/E table library

Notes:

1. Include GIM.SGIMTENU and the SMPTABL data set in the ISPTLIB concatenation.

2. Use the ISPCTL1 and ISPCTL2 files to generate JCL for submitted SMP/E jobs. The SMP/E job submit facility lets you browse and edit this JCL. You can omit these files from your logon procedure and let ISPF automatically allocate them as needed.

   To save the input JCL generated by the dialogs, either:
   - Use EDIT CREATE while in the generated JCL to save it in another (permanent) data set, or
   - Allocate a permanent sequential data set to ISPCTL1 (LRECL=80, RECFM=FB) **before** you enter the SMP/E dialogs.
3. Allocate a single, installation-wide table data set to the ISPTLIB and SMPTABL DD statements.
   - SMP/E uses this table data set to save process status information for the SYSMOD management dialogs.
   - The data set must be a partitioned data set (LRECL=80, RECFM=FB). Because the data set is also in the concatenation of ISPTLIB, make the block size compatible with the block size of the corresponding ISPF data sets.
   - For more information about the SMPTABL data set, see "SMPTABL Space Allocation".

SMPTABL Space Allocation

The amount of space required for SMPTABL depends on the number of concurrent processes you want to support for the SMP/E SYSMOD management dialog and on the amount of data that is saved for each process. Use Table 8 as a guide.

Table 8. SMPTABL Data Set Allocations

Number of Processes	BLKSIZE	Number of Blocks	Directory Blocks
16	8800	60	4
32	8800	120	8

Sample Logon Procedure

Figure 32 is an example of a logon procedure that concatenates the libraries for the SMP/E dialogs.

Note: Depending on the system you are connecting the dialogs to, you may need to change your logon procedure, a CLIST, or both. It is common practice to have a logon procedure call a CLIST every time a user logs on. This is normally done so that minor changes to concatenations do not require changes to the logon procedure, or so users with the same logon procedure can have the CLIST perform different allocations from the ones performed for other users.

If you make changes **only** to your logon procedure and the procedure calls a CLIST that changes a concatenation, you may end up missing changes you made in your logon procedure. In this case, instead of (or in addition to) updating the concatenations in your logon procedure, you should update the concatenations in the CLIST that is called.

```
//SMPE     EXEC PGM=IKJEFT01,ROLL=(NO,NO),DYNAMNBR=99
//SYSPROC  DD DSN=GIM.SGIMCLS0,DISP=SHR     /* SMP/E CLISTs. */
//         DD DSN=ISR.V2R3M0.ISRCLIB,DISP=SHR
//SYSHELP  DD DSN=SYS1.HELP,DISP=SHR
//SYSPRINT DD TERM=TS
//SYSIN    DD TERM=TS
//*************************************************************
//*
//* ISPF temporary data sets (no ISPCTL1 or ISPCTL2)
//*
//*************************************************************
//ISPLST1 DD DISP=NEW,UNIT=SYSALLDA,SPACE=(CYL,(1,1)),
//        DCB=(LRECL=121,BLKSIZE=0,RECFM=FBA)
//ISPLST2 DD DISP=NEW,UNIT=SYSALLDA,SPACE=(CYL,(1,1)),
//        DCB=(LRECL=121,BLKSIZE=0,RECFM=FBA)
//*************************************************************
//*
//* ISPF/SMPE load modules, panels, skeletons, messages,
//*           tables
//*    NOTE:  SMPTABL is a single installation-wide
//*           ISPF table data set shared by all SMP/E
//*           users. It is initially allocated as an empty PDS
//*           LRECL=80, RECFM=FB, BLKSIZE= (compatible with
//*           ISP.V2R3M0.ISRTLIB).
//*
//*************************************************************
//ISPLLIB DD DSN=GIM.SGIMLMD0,DISP=SHR     /* SMP/E LMODs.  */
//ISPPLIB DD DSN=GIM.SGIMPENU,DISP=SHR     /* SMP/E panels. */
//        DD DSN=ISR.V2R3M0.ISRPLIB,DISP=SHR
//        DD DSN=ISP.V2R3M0.ISPPLIB,DISP=SHR
//ISPSLIB DD DSN=GIM.SGIMSENU,DISP=SHR   /* SMP/E skeletons.*/
//        DD DSN=ISR.V2R3M0.ISRSLIB,DISP=SHR
//        DD DSN=ISP.V2R3M0.ISPSLIB,DISP=SHR
//ISPMLIB DD DSN=GIM.SGIMMENU,DISP=SHR   /* SMP/E messages. */
//        DD DSN=ISR.V2R3M0.ISRMLIB,DISP=SHR
//        DD DSN=ISP.V2R3M0.ISPMLIB,DISP=SHR
//ISPTLIB DD DSN=GIM.SGIMTENU,DISP=SHR     /* SMP/E tables. */
//        DD DSN=ISR.V2R3M0.ISRTLIB,DISP=SHR
//        DD DSN=ISP.V2R3M0.ISPTLIB,DISP=SHR
//        DD DSN=user-defined name,DISP=SHR /* SMP/E tables.*/
//SMPTABL DD DSN=user-defined name,DISP=SHR /* SMP/E tables.*/
```

Figure 32. Sample Logon Procedure That Concatenates SMP/E and ISPF Libraries

Connect the Dialogs to ISPF

You can get to the SMP/E dialogs from the SMP/E primary option menu. To provide access to that menu, you must connect the dialogs to ISPF. Sample ISPF panels are provided with z/OS to enable panels for most z/OS elements, including SMP/E. These panels are in the SISPPENU data set after APPLY processing. The panels supplied are:

ISR@390 This is an ISPF Primary Options Menu. It is identical to the ISR@PRIM Primary Options Menu, except that it includes the additional options 12 and 13, which point to the next two panels.

ISR@390S This is a secondary panel, with options used by system programmers and administrators, including SMP/E.

ISR@390U This is a secondary menu panel, including the options used by most ISPF users.

Connecting SMP/E Dialogs to ISPF

Use one of the methods documented in the *Program Directory* for this level of SMP/E to use ISR@390 instead of ISR@PRIM. The SMP/E dialogs will then be accessible through panel ISR@390S.

Customize the SMP/E Dialogs

After you install the SMP/E dialogs, you can change the default values they use.

When you select option 0 (SETTINGS) on the SMP/E Primary Option Menu, SMP/E displays panel GIM@PARM, which allows you to

- Modify the values for allocating temporary SMP/E utility (SYSUT*n*) and SMP/E work (SMPWRK*n*) data sets . The options you specify are saved permanently in the ISPF profile pool for use later by other SMP/E dialog processes.
- Specify a data set name to be used by SMP/E for the SMPPARM data set during background execution. The SMPPARM data set is used to define exit routines and specify allocation information used by SMP/E processing. If a data set name is specified on panel GIM@PARM, SMP/E generates an SMPPARM DD statement in all JCL jobs created by the SMP/E dialogs that invoke SMP/E.

```
GIM@PARM                      SMP/E DIALOG SETTINGS
===>

Reset to use SMP/E's default settings? ===> NO  (Yes or No)

Enter or verify the information below used to allocate temporary data sets.
These temporary data sets contain generated JCL jobs, output for jobs which
have completed execution, and MCS entries for viewing:

   UNIT   ===> SYSALLDA
   VOLUME ===>

Enter or verify the information below used to create DD statements in
generated JCL jobs:

   Space for SMP/E Utility Work data sets (SYSUTn):
             Block Size  Primary  Secondary
     SYSUT1     3120       380       760
     SYSUT2     3120       380       760
     SYSUT3     3120       380       760
     SYSUT4     3120        38       100

   Space for SMP/E Work data sets (SMPWRKn):
             Block Size  Primary  Secondary  Directory Blocks
     SMPWRK1    3120       364       380          500
     SMPWRK2    3120       364       380          500
     SMPWRK3    3120       364       380          500
     SMPWRK4    3120       364       380          500
     SMPWRK6    3120       364       380          500

   Unit for SYSUTn and SMPWRKn data sets:
     UNIT ===> SYSDA

   Data set name to use for SMPPARM.  If a name is specified, an SMPPARM DD
   statement will be generated.
     DSN  ===>

Enter or verify the information below used to allocate permanent SMPWRK3
data sets created and used by the SYSMOD Management dialogs:

   UNIT   ===>
   VOLUME ===>
   PREFIX ===>          (TSO Prefix is used if no prefix is specified)

To save the changes, press ENTER .
To ignore the changes, enter END .
```

JOB Statement Customization

If you have never entered a JOB statement on panels GIMCGSUB, GIMRCSUB, or GIMSB01, then SMP/E primes those panels with the following default JOB statement:

```
//useridA  JOB (ACCOUNT),'NAME'
//*
//*
//*
```

This initial default JOB statement is not customizable, but when you enter a JOB statement on panels GIMCGSUB, GIMRCSUB, or GIMSB01, the statement you enter is saved in your ISPF profile pool and will be used as the new default when you use those panels again.

Setting Up SMP/E for Easier Operation

SMP/E provides several optional facilities that you can use to make SMP/E operations easier and more efficient. To take advantage of these facilities, you must set up a few SMP/E options. Normally, these set up procedures need only be done once.

The major tasks are:
- Specifying SMP/E OPTIONS entry
- Specifying link edit utility output DDDEF entries
- Specifying automatic cross-zone requisite checking

Recommended Values for OPTIONS Entry

IBM recommends the following OPTIONS entry values:

Recommended Value	Purpose
MSGFILTER(YES)	MSGFILTER(YES) causes SMP/E to filter the messages it writes to SMPOUT during APPLY, ACCEPT, and RESTORE command processing. When SMP/E filters messages, most non-critical informational messages are not written to SMPOUT. The result is less output to read through when it is necessary to investigate an SMP/E operation. MSGFILTER(NO) is the default.
MSGWIDTH(80)	MSGWIDTH(80) causes SMP/E to format its messages to an 80 character width. This makes online viewing simpler by eliminating the need to scroll right to view the entire message text. MSGWIDTH(120) is the default.
RECZGRP	Often the RECEIVE command will receive a PTF that has already been accepted and purged from the global zone and SMPPTS data set. There is no need to receive such PTFs and they only add to the space used by the SMPPTS. To prevent RECEIVE from receiving such PTFs, you need to tell SMP/E what dlib zones to check when determining if a PTF has already been accepted. You can specify the list of dlib zones using the RECEIVE Zone Group (RECZGRP) subentry in an OPTIONS entry. The RECZGRP subentry allows you to set a policy and specify the list of zones once. This list is then used for all future RECEIVE operations whenever the OPTIONS entry is active. With the list of dlib zones set, during RECEIVE processing, SMP/E will check each of the zones specified first before receiving a PTF. If that PTF is accepted in any of the specified zones, the PTF will not be received again.
RETRYDDN(ALL)	RETRYDDN(ALL) causes SMP/E to compress out-of-space libraries and retry processing after an *x*37 abend. When you use this option, make sure you are **not** updating production data sets.

Note: Do not specify a PEMAX value. Allow SMP/E to use its default value of 25,000.

Sample UCLIN Job

Here is a sample UCLIN job to build an OPTIONS entry with the recommended values:

```
//job      JOB accounting info...
//step     EXEC PGM=GIMSMP
//SMPCSI   DD DSN=smp.global.csi,DISP=SHR
//SMPCNTL  DD *
  SET BDY(GLOBAL).
  UCLIN.
  ADD OPTIONS(OPTENT)
      MSGFILTER(YES)
      MSGWIDTH(80)
      RETRYDDN(ALL)
      RECZGRP( zosdlib
               os390dlib
               jes2dlib
               jes3dlib
               cicsdlib
               db2dlib
               imsdlib ).
  ENDUCL.
/*
```

Activating the OPTIONS Entry

After the OPTIONS entry has been defined, IBM recommends that you make it active by defining it as the default OPTIONS entry for the global, target, and DLIB zones. Otherwise, you must specify it on the SET command before using any other SMP/E command.

Recommended Link Edit Utility Output DDDEF Entries

To exploit utility multi-tasking in SMP/E, ensure the ddname that is to contain the link edit utility output is defined with a DDDEF entry that identifies a SYSOUT class. SMP/E's default ddname for utility output is SYSPRINT, but can be changed using the PRINT subentry of the LKED UTILITY entry. When multi-tasking, SMP/E will invoke multiple instances of the link edit utility at the same time, thus decreasing the total time required to complete an APPLY, ACCEPT, or RESTORE command. If you do not define the print ddname using a DDDEF entry, or if the DDDEF identifies something other than a SYSOUT class, SMP/E will not multi-task link edit utility operations.

Specifying Automatic Cross-Zone Requisite Checking

The installation of software service often requires the synchronization of service levels across multiple SMP/E zones. For example, service for software in the MVS zone may require related service for the JES2, CICS, DB2, and other zones to permit all software within the system image to operate properly. To help ensure proper synchronization across zones, you can tell SMP/E to automatically check for cross-zone requisites during APPLY, ACCEPT, and RESTORE command processing.

To enable automatic cross-zone requisite checking, you must tell SMP/E which zones contain software to be checked for requisites. The set of zones identified for cross-zone requisite checking is called the *zone group*. SMP/E provides two methods to identify the zones within the zone group:

1. Define a default zone group
2. Specify the zones directly on the APPLY, ACCEPT, or RESTORE command.

Defining a Default Zone Group

You can define a default zone group by creating a ZONESET entry that contains the XZREQCHK(YES) subentry and the list of zones to be included in the default zone group. SMP/E will use this default zone group to determine which zones to check for requisites whenever the APPLY, ACCEPT, or RESTORE commands process a zone named in this ZONESET. To create such a ZONESET, use the SMP/E Administration Dialogs or use the UCLIN command, as in this example:

```
//job      JOBaccounting info...
//step     EXEC PGM=GIMSMP
//SMPCSI   DD DSN=smp.global.csi,DISP=SHR
//SMPCNTL  DD *
  SET BDY(GLOBAL).
  UCLIN.
    ADD ZONESET(ZONEGRP)
        XZREQCHK(YES)
/* use this ZONESET for cross-zone req checking */
        ZONE(zostgt   zosdlib
             os390tgt os390dlib
             jes2tgt  jes2dlib
             jes3tgt  jes3dlib
             cicstgt  cicsdlib
             db2tgt   db2dlib
             imstgt   imsdlib).
  ENDUCL.
/*
```

The ZONESET should contain the names of all the zones to be checked for cross-zone requisites. Once the ZONESET is created and the XZREQCHK(YES) subentry is set, the zones defined in the ZONESET are used as the default zone group any time the APPLY, ACCEPT, or RESTORE commands process any zone found in the ZONESET. For example, if an APPLY command is initiated for the *cicstgt* zone, all zones found in the ZONESET entry named ZONEGRP are used for the zone group.

Specifying the Zone Group on a Command

If you don't have a default zone group defined, or you want to use a different set of zones for the zone group, you can specify the zones on the APPLY, ACCEPT, or RESTORE command using the XZGROUP operand. This is simply a matter of specifying the zones to be checked for cross-zone requisites, as shown in this example:

```
//job      JOB accounting info...
//step     EXEC PGM=GIMSMP
//SMPCSI   DD DSN=smp.global.csi,DISP=SHR
//SMPCNTL  DD *
  SET BDY(zostgt).
  APPLY SOURCEID(HIPER)
        CHECK
        XZGROUP(os390tgt,jes2tgt,jes3tgt,cicstgt,db2tgt,imstgt)
        BYPASS(HOLDSYS).
/*
```

Define a ZONEINDEX for Each Zone

Each of the zones specified in a ZONESET or on the XZGROUP operand must be defined by a ZONEINDEX in the current global zone, even if the zones are already defined in another global zone (more than one global zone may contain a ZONEINDEX for the same target or dlib zone). This allows the APPLY, ACCEPT, and RESTORE commands initiated from the current global zone to access the specified zones. To add ZONEINDEX subentries for each of the zones, use the SMP/E Administration Dialogs or use the UCLIN command, as in this example:

```
//job      JOB accounting info...
//step     EXEC PGM=GIMSMP
//SMPCSI   DD DSN=smp.global.csi,DISP=SHR
//SMPCNTL  DD *
  SET BDY(GLOBAL).
  UCLIN.
    ADD GLOBALZONE ZONEINDEX(
        (zostgt,zos.target.csi,TARGET)
        (zosdlib,zos.dlib.csi,DLIB)
        (os390tgt,os390.target.csi,TARGET)
        (os390dlib,os390.dlib.csi,DLIB)
        (jes2tgt,jes2.target.csi,TARGET)
        (jes2dlib,jes2.dlib.csi,DLIB)
        (jes3tgt,jes3.target.csi,TARGET)
        (jes3dlib,jes3.dlib.csi,DLIB)
        (cicstgt,cics.target.csi,TARGET)
        (cicsdlib,cics.dlib.csi,DLIB)
        (db2tgt,db2.target.csi,TARGET)
        (db2dlib,db2.dlib.csi,DLIB)
        (imstgt,ims.target.csi,TARGET)
        (imsdlib,ims.dlib.csi,DLIB)
        ).
  ENDUCL.
/*
```

Cross-Zone Requisite Checking

Whether you define a default zone group or specify a zone group on the APPLY, ACCEPT, and RESTORE command, SMP/E will determine during command processing whether any cross-zone requisites are unsatisfied. Cross-zone requisites are caused by ++IF statements, where a SYSMOD containing a ++IF statement resides in one zone and the function identified on the ++IF resides in another zone. If the requisite identified on the ++IF statement does not reside in the same zone as the identified function, then the condition is not satisfied.

Unsatisfied cross-zone requisite conditions will cause APPLY, ACCEPT, and RESTORE command processing to fail for the SYSMOD containing the ++IF statement. Processing will continue to fail until the requisite is satisfied in the other zone, unless the BYPASS(XZIFREQ) operand is specified on the command.

Bypassing Unsatisfied Cross-Zone Requisites

The BYPASS(XZIFREQ) operand on the APPLY, ACCEPT, and RESTORE commands tells SMP/E to continue processing the command even if missing cross-zone requisites are detected. SMP/E warning messages will be issued to identify the missing cross-zone requisites.

```
//job      JOB accounting info...
//step     EXEC PGM=GIMSMP
//SMPCSI   DD DSN=smp.global.csi,DISP=SHR
//SMPCNTL  DD *
  SET BDY(zostgt).
  APPLY SOURCEID(HIPER)
        CHECK
        BYPASS(HOLDSYS
               XZIFREQ).
/*
```

Note: This example assumes a default zone group has been defined and will therefore be used during APPLY command processing.

You can be broad or very granular in the specification of what cross-zone requisites to bypass. You can indicate all cross-zone requisites are to be bypassed (as in the previous example), you can indicate that specific cross-zone requisite SYSMODs are to be bypassed, or you can indicate that only specific cross-zone requisite

SYSMODs from specific zones are to be bypassed. Details of the BYPASS(XZIFREQ) operand and processing can be found in *SMP/E Commands*.

Resolving Cross-Zone Requisites

If cross-zone requisites are bypassed and therefore cause unsatisfied cross-zone requisites, you must resolve those unsatisfied requisites. To do this, you need to APPLY or ACCEPT those requisites to the appropriate zones. To aid in this task, SMP/E provides a method to identify missing cross-zone requisite SYSMODs and make them candidates for APPLY and ACCEPT processing to resolve missing cross-zone requisites.

In order to select cross-zone requisite SYSMODs to be installed in a particular zone, the XZREQ operand can be used on the APPLY and ACCEPT commands. The XZREQ operand causes SMP/E to search the zones in the zone group for unsatisfied cross-zone requisites. If any are found which can be satisfied by installing a requisite SYSMOD to the current zone, those SYSMODs are made candidates for the APPLY and ACCEPT commands. Here is an example:

```
//job      JOBaccounting info...
//step     EXEC PGM=GIMSMP
//SMPCSI   DD DSN=smp.global.csi,DISP=SHR
//SMPCNTL  DD *
  SET BDY(cicstgt).
  APPLY CHECK
        BYPASS(HOLDSYS)
        XZREQ.
/*
```

Note: This example assumes a default zone group has been defined and will therefore be used during APPLY command processing.

Using the XZREQ operand identifies and installs the needed cross-zone requisites. You can also use the REPORT CROSSZONE command to identify the needed cross-zone requisites. See The REPORT CROSSZONE Command in *SMP/E Commands* for details.

Defining the Information Needed to Invoke SMP/E

There are several ways to call SMP/E after it has been installed:

- Use the SMP/E dialogs.
- Submit a background job that calls GIMSMP, the program name for SMP/E. This job can call SMP/E either directly or in a cataloged procedure.

This section describes the types of information you need to provide if you use a cataloged procedure to invoke SMP/E. It discusses the following:

- Required JCL statements
- A sample cataloged procedure for SMP/E

Required JCL Statements

Unless you are using the SMP/E dialogs, you must provide the following JCL statements to invoke SMP/E:

- A JOB statement
- An EXEC statement
- DD (data definition) statements

JOB Statement

The **JOB statement** describes your installation-dependent parameters. The JOB statement (or the EXEC statement, or both) can also include the REGION

parameter to set the size of the region in which SMP/E runs. For details, see *z/OS MVS JCL User's Guide*, SA22-7598 or *z/OS MVS JCL Reference*, SA22-7597.

Note: To enable the SMP/E job step to get the maximum space above 16 megabytes. you can specify REGION=0M. Or, if you prefer, you can specify a more specific region size.

EXEC Statement

The **EXEC statement** must specify PGM=GIMSMP or the name of your cataloged procedure for calling SMP/E. (For an example of a cataloged procedure, see "Sample Cataloged Procedure for SMP/E" on page 78.) The following can be specified in the EXEC statement PARM parameter:

CSI= *dsname*

where *dsname* is the name of the CSI data set containing the global zone. (This data set is also known as the *master CSI*.) This parameter is used to enable SMP/E to allocate the master CSI data set dynamically.

Note: If there is an SMPCSI DD statement, the **CSI=***dsname* operand is not allowed. If both are specified, SMP/E does **not** run.

DATE= *date*

where *date* can be one of the following:

U or IPL	To use the IPL date of the system.
REPLY	To request the date from the operator. As a result, SMP/E issues message GIM399I.
yyddd	To specify a specific date, where *yy* is the year and *ddd* is the day of the year (the Julian date).

If **DATE** is not specified, the IPL date of the system is used.

PROCESS=WAIT or
PROCESS=END

The PROCESS parameter is used to control how long a job should wait if a CSI or PTS data set is not immediately available because it is currently being used either by another job or by a dialog.

- WAIT causes the job to wait until the data set is available. A message is issued to the system operator every 30 minutes while the job is waiting.
- END causes the job to wait for 10 minutes. If the data set is still not available after the 10-minute wait, the command requiring the data set is stopped.

If **PROCESS** is not specified, the default is PROCESS=WAIT.

For more information on obtaining and sharing CSI data sets, see "Sharing SMP/E Data Sets" in *SMP/E Commands*.

Processing of the PTS data set is also affected by the WAITFORDSN value specified in its DDDEF entry. WAITFORDSN determines whether SMP/E should wait to allocate a data set that is not immediately available. If the DDDEF entry specifies WAITFORDSN=NO (or lets this value default to NO) and the data set is not available, allocation of the data set fails, regardless of the PROCESS value specified on the EXEC statement. If WAITFORDSN=NO, SMP/E does not wait to retry allocation of the data set.

For example, suppose a PTS with a disposition of OLD is already being used by a job, and a second job tries to access the same PTS data set by allocating it through a DDDEF entry. The DDDEF entry used by the second job for the PTS specifies WAITFORDSN=NO. As a result, allocation of the PTS fails for the second job.

DD Statements

DD statements define the data sets that can be used in SMP/E processing. For information on the data sets required for each command, see the chapters on individual SMP/E commands in *SMP/E Commands*.

Note: You can use DDDEF entries, rather than DD statements, to allocate many of the necessary data sets. For more information, see "How to Dynamically Allocate Data Sets to Be Used During SMP/E Processing" on page 58.

Sample Cataloged Procedure for SMP/E

Figure 33 on page 80 is a sample cataloged procedure, SMPPROC, that can be used to run SMP/E. The numbers to the left of the statements correspond to the notes that follow the example. When you write a cataloged procedure for SMP/E, remember the following:

- Tailor your own cataloged procedure to fit your system and processing requirements.
- You may want to use a single procedure for all SMP/E processing, or you may want to define multiple procedures for specific SMP/E commands and include in each one just those DD statements required for that command. For example, a special procedure for RECEIVE might include the SMPPTFIN DD statement but no DD statements for the target and distribution libraries.

  Note: The SYSLIB concatenations for APPLY and ACCEPT should point to different libraries.
- Most of the data sets in the cataloged procedure can be allocated without DD statements. If you use the methods described for the data sets listed below, you may not need a cataloged procedure.
  - **Master CSI data set.** The master CSI data set can be specified on the CSI= parameter of the EXEC statement for GIMSMP, rather than on the SMPCSI DD statement. For more information about parameters you can specify on the EXEC statement, see "Required JCL Statements" on page 76.
  - **Target and distribution zones.** CSI data sets for target and distribution zones are normally dynamically allocated with zone indexes in the global zone. If you want to use the batch local shared resources (BLSR) subsystem, you must supply your own JCL statements. For examples of defining zone indexes and of specifying JCL for BLSR, see the SMPCSI section in *SMP/E Reference*.
  - **Other data sets.** Other data sets in the cataloged procedure can be defined with DDDEF entries. When you use DDDEF entries, only the data sets SMP/E needs for a particular run are allocated.

    When you use DD statements, all the data sets defined must be online and allocated. Therefore, you might want to use a combination of DDDEF entries and a cataloged procedure shorter than the one in Figure 33 on page 80. For more information about DDDEF entries, see *SMP/E Reference*.
- Although they are not shown in the sample cataloged procedure, the following DD statements may also be required:
  - An SMPCNTL DD statement, pointing to the commands that SMP/E processes, is required by all commands.

  - An SMPPTFIN DD statement, pointing to the source of the SYSMODs that are processed, is required by the RECEIVE command.
  - An SMPHOLD DD statement, pointing to the source of the HOLDDATA that is processed, is required by the RECEIVE command.
  - An SMPPTS DD statement should be coded with **DISP=SHR**. This allows concurrent jobs to share the PTS as much as possible. For more information on how SMP/E shares data sets, see Sharing SMP/E Data Sets in *SMP/E Commands*.
  - The SMPLTS data set is required when processing load modules with CALLLIBS.
  - An SMPMTS DD statement is required for changes to macros that do not reside in a target library.
  - An SMPSTS DD statement is required for changes to source code that does not reside in a target library.
  - If any of the SYSMODs being installed contain elements packaged with the LKLIB operand, a DD statement for the ddname specified on that operand is required by the APPLY and ACCEPT commands.
  - If any of the SYSMODs being installed contain elements or JCLIN packaged with the TXLIB operand, a DD statement for the ddname specified on that operand is required by the APPLY and ACCEPT commands.
- If any of the required data sets (such as the SMPPTFIN) are not defined in the cataloged procedure or by DDDEF entries, they must be specified in the JCL used to call SMP/E.
- For more information about the data sets required by each SMP/E command, see *SMP/E Commands*.

Sample Cataloged Procedure

```
    //SMPPROC  PROC
    //SMP      EXEC PGM=GIMSMP
    //* -----  SYSOUT data sets --------------------------------
    //SMPOUT   DD SYSOUT=A
    //SMPRPT   DD SYSOUT=A
    //SMPLIST  DD SYSOUT=A
    //SMPSNAP  DD SYSOUT=A
    //SMPDEBUG DD SYSOUT=A
    //SYSPRINT DD SYSOUT=A
1   //SMPPUNCH DD SYSOUT=B
    //*------  SMP/E data sets ---------------------------------
    //SMPLOG   DD DSN=SYS1.SMPLOG,DISP=MOD
    //SMPLOGA  DD DSN=SYS1.SMPLOGA,DISP=MOD
2   //* -----  Master CSI  -------------------------------------
    //SMPCSI   DD DSN=SMPE.SMPCSI.CSI,DISP=SHR
    //SMPDATA1 DD DSN=MVSTGT1.SMPDATA1,DISP=MOD
    //SMPDATA2 DD DSN=MVSTGT1.SMPDATA2,DISP=MOD
3   //* -----  SMP/E temporary data sets------------------------
    //SMPWRK1  DD UNIT=SYSDA,SPACE=(CYL,(2,1,5)),DISP=(,DELETE),
    //            DCB=BLKSIZE=6160
    //SMPWRK2  DD UNIT=SYSDA,SPACE=(CYL,(2,1,5)),DISP=(,DELETE),
    //            DCB=BLKSIZE=6160
4   //SMPWRK3  DD DSN=data set name,
                UNIT=SYSDA,SPACE=(CYL,(2,1,5)),DISP=(,CATALOG),
    //            DCB=BLKSIZE=3120
    //SMPWRK4  DD UNIT=SYSDA,SPACE=(CYL,(2,1,5)),DISP=(,DELETE),
    //            DCB=BLKSIZE=3120
    //SMPWRK6  DD UNIT=SYSDA,SPACE=(CYL,(2,1,5)),DISP=(,DELETE),
    //            DCB=BLKSIZE=6160
    //* -----  Utility data sets -------------------------------
    //SYSUT1   DD UNIT=SYSDA,SPACE=(CYL,(2,1)),DISP=(,DELETE)
    //SYSUT2   DD UNIT=SYSDA,SPACE=(CYL,(2,1)),DISP=(,DELETE)
    //SYSUT3   DD UNIT=SYSDA,SPACE=(CYL,(2,1)),DISP=(,DELETE)
    //SYSUT4   DD UNIT=SYSDA,SPACE=(TRK,(2,1)),DISP=(,DELETE)
    //* -----  Assembler SYSLIB data set -----------------------
5   //SYSLIB   DD DSN=data set name,DISP=SHR
                   •  •  •
    //* -----  Target libraries --------------------------------
    //LINKLIB  DD DSN=SYS1.LINKLIB,DISP=OLD
                   •  •  •
    //* -----  Distribution libraries --------------------------
    //AOSC5    DD DSN=SYS1.AOSC5,DISP=OLD
                   •  •  •
    //         PEND
```

Figure 33. Sample SMP/E Cataloged Procedure

1. **SMPPUNCH** is required for the GENERATE, REPORT, and UNLOAD commands. Because it may have a high level of output, SMPPUNCH should be directed to disk or tape.

2. **SMPCSI** DD statements should be coded with **DISP=SHR**. This allows SMP/E to share the CSI data sets as much as possible. If **DISP=OLD** is specified, no data-set sharing is attempted. For more information on how SMP/E shares data sets, see Sharing SMP/E Data Sets in *SMP/E Commands*.

3. **SMPWRK1–SMPWRK6** show only sample sizes for the data sets. The actual size required depends on the number of SYSMODs being processed and the number of elements within those SYSMODs.

4. **SMPWRK3** can be permanently allocated in order to reuse assemblies. For more information, see the description of the REUSE operand in The APPLY Command in *SMP/E Commands*.

5. **SYSLIB** concatenation depends on how you intend to use the distribution

libraries. For details on which data sets to include and in what order, see "How to Determine the Appropriate SYSLIB Concatenation".

If you use a different SYSLIB concatenation for APPLY and ACCEPT and prefer to use a SYSLIB DD statement, you should have at least two procedures. If you use DDDEFs to point to the different library concatenations, you can use one procedure. You can modify the examples to use the appropriate procedure.

The following job uses the cataloged procedure in Figure 33 on page 80 to call SMP/E.

```
//SMPJOB   JOB 'accounting info',MSGLEVEL=(1,1)
//SMPSTEP  EXEC SMPPROC
//SMPPTFIN DD  ... points to the file or data set that contains
//*               the SYSMODs to be received
//SMPHOLD  DD  ... points to the file or data set that contains
//*               the HOLDDATA to be received
//SMPTLIB  DD  UNIT=SYSALLDA,VOL=SER=SMPVOL
//SMPCNTL  DD  *
  SET      BDY(GLOBAL)        /* Set to global zone       */.
  RECEIVE  SYSMOD             /* receive SYSMODS and      */
           HOLDDATA           /* HOLDDATA                 */
           SOURCEID(MYPTFS)   /* Assign a source ID       */
                              /*                          */.
  LIST     MCS                /* List the cover letters   */
           SOURCEID(MYPTFS)   /* for the SYSMODs          */
                              /*                          */.
  SET      BDY(TARGET1)       /* Set to target zone       */.
  APPLY    SOURCEID(MYPTFS)   /* Apply the SYSMODs        */
                              /*                          */.
  LIST     LOG                /* List the target zone log */.
/*
```

How to Determine the Appropriate SYSLIB Concatenation

The recommended method for determining the appropriate SYSLIB concatenation treats the distribution libraries as totally separate from the target libraries.

Note: The example shown is for processing a zone for SREL Z038 containing the z/OS base control program (BCP). For other zones, follow the recommendations for the products residing in those zones.

Treating the distribution libraries as totally separate from the target libraries ensures that only the latest tested version of a macro is used during an assembly. Thus, the SYSLIB concatenation at ACCEPT is different from that at APPLY.

The SMPMTS data set contains macros from SYSMODs that are applied. Therefore, the proper SYSLIB concatenation for APPLY processing includes the SMPMTS data set, as is shown in Figure 34 on page 82.

Sample Cataloged Procedure

```
//* ------ Include SMPMTS first
//* ------ followed by all macro target libraries
//* ------ followed by all distribution
//*        libraries
//SYSLIB   DD  DSN=SYS1.SMPMTS,DISP=OLD
//         DD  DSN=SYS1.MACLIB,DISP=OLD
//         DD  DSN=SYS1.MODGEN,DISP=OLD
//* ------ IF YOU NEED TO ASSEMBLE JES SOURCE MODULES:
//* ------ either HASPSRC (JES2) or JES3MAC (JES3)
//         DD  DSN=SYS1.HASPSRC,DISP=OLD
//         DD  DSN=SYS1.SISTMAC1,DISP=OLD
//         DD  DSN=SYS1.ATSOMAC,DISP=OLD
//         DD     •
//         DD     • other macro target libraries
//         DD     •
//         DD     •
//         DD     • any macro distribution libraries needed
//         DD     •
```

Figure 34. APPLY SYSLIB Concatenation: APPLY Different from ACCEPT

During ACCEPT processing, the macros in the SMPMTS and in the target macro libraries are not considered to have been tested. The SMPMTS is, therefore, not concatenated. Figure 35 shows the proper SYSLIB concatenation for ACCEPT.

```
//* ------ Include only macro distribution
//*        libraries
//         DD  DSN=SYS1.AMACLIB,DISP=OLD
//SYSLIB   DD  DSN=SYS1.AMODGEN,DISP=OLD
//* ------ IF YOU NEED TO ASSEMBLE JES SOURCE MODULES:
//* ------ either HASPSRC (JES2) or JES3MAC (JES3)
//         DD  DSN=SYS1.HASPSRC,DISP=OLD
//         DD  DSN=SYS1.AISTMAC1,DISP=OLD
//         DD  DSN=SYS1.ATSOMAC,DISP=OLD
//         DD     •
//         DD     • other macro distribution libraries
//         DD     •
```

Figure 35. ACCEPT SYSLIB Concatenation: APPLY Different from ACCEPT

Defining Exit Routines

There are two types of exit routines you can define to tailor SMP/E processing:

- The **RECEIVE exit routine** enables you to scan statements in the SMPPTFIN data set during RECEIVE processing.
- The **retry exit routine** enables you to control retry processing when a data set runs out of space during RETRY can be specified on the ACCEPT, APPLY, LINK LMODS, LINK MODULE, and RESTORE commands. processing. (In retry processing, the data set is compressed and the utility that failed is called again.)

For more details, see the “SMP/E Exit Routines” chapter, in *SMP/E Reference*.

Chapter 4. Installing a New Function

This chapter discusses the method you use to install a new function. It describes:
- Sources of new functions and sources of installation information
- An example of installing a function

Introduction

The primary purpose of SMP/E is to help you install SYSMODs in your target and distribution libraries. To do this, SMP/E provides three basic commands: RECEIVE, APPLY, and ACCEPT.

This chapter summarizes the general steps you follow to install a function. You should look at the installation materials that arrive with a function to find out about special requirements and procedures for installing the function. Table 9 lists sources of new functions and places where you can find information for installing the new functions.

Table 9. Sources for Functions and Their Installation Information

Product Delivery Vehicle	Products and Service You Get	Installation Materials You Can Use
CBPDO	Products and service (not integrated) on a single logical tape	Sample jobs to receive products and service Program directories for the products you ordered Installation manuals for the products you ordered
Independent product	Product tape	Program directory for the product Installation manual for the product (if one is provided)

RECEIVE-APPLY-ACCEPT Method

RECEIVE-APPLY-ACCEPT is the **standard** installation method. It uses SMP/E RECEIVE, APPLY, and ACCEPT commands to install functions onto a subsystem. Usually, you do not have to do any special processing outside SMP/E to install your functions. The JCLIN needed to set up the load modules for the function is sent along with the function.

In this method of installation, you:

1. Use the RECEIVE command to get the SYSMODs from the input data set and put them in the SMP/E data sets (the PTS and global zone within the CSI).
2. Use the APPLY command to install the SYSMODs in the target system libraries; then test them as required.
3. Use the ACCEPT command to install the SYSMODs into the distribution libraries.

Using the Standard RECEIVE-APPLY-ACCEPT Method

This section describes the standard process for using the RECEIVE, APPLY, and ACCEPT commands to install a function.

Note: You can use either the SMP/E dialogs or JCL jobs to receive, apply, and accept functions. The basic steps to follow are the same. If you have access to the SMP/E dialogs, you should use them. Otherwise, you can use the steps described in this chapter as examples.

Preparing Your System

Before you start doing any operations on the product function or service tape you have received, there is work you should do to your system to make sure it is ready and to be certain you can recover in case of a serious failure during installation.

Following are some of these steps:

1. **Read the documentation** for your new product. This includes the program directory and, if provided, an installation guide. Also check the IBM Preventive Service Planning (PSP) files for the latest information about the product. This is important because there may be a PTF for the product that is not included in an ESO, or one of the PTFs may contain an error you should know about.
2. When you order a product, you should **update the FMID list** in the global zone with the FMIDs of the products you will be receiving. (Check the program directory for this information.) This enables you to receive any preventive service shipped between the time you order the product and the time you install it.
3. **Read the program directory.** It tells you which libraries are affected, whether any existing libraries must be expanded and by how much, and whether any new libraries are required.
4. **Prepare the target and distribution libraries.** If these libraries are properly prepared before you apply or accept a SYSMOD, very little time is lost if an error occurs.

   List the VTOC of the target and distribution packs. This shows you which data sets are into their secondary extents, or are too full to contain additional elements that may be applied or accepted. If you are unsure how large a data set will grow, you may want to check full data sets against the SYSMODs you will be processing.

   Partitioned data sets with a high percentage of their space used can be compressed by use of IEBCOPY. If it looks as if more space may be needed even after the compression, allocate a larger data set and copy the nearly full data set into it; then delete the old data set. Rename the new one properly and, if it has had to be allocated on a different pack, update any procedure necessary with the new VOLUME data.

   This preparation is time-consuming but takes less time and work than recovering from an out-of-space abend (E37, B37, and so on).

   SMP/E command operands can also help you handle out-of-space abends.

   - The COMPRESS operand tells SMP/E to compress the data sets **before** they are updated; this can help you avoid an *x*37 ABEND. For more information about the COMPRESS operand, see the *SMP/E Commands* manual.
   - The RETRY(YES) operand tells SMP/E to attempt recovery **after** an *x*37 ABEND occurs by compressing the affected data sets and retrying the failing utility. If you still need space after SMP/E's initial retry attempt and input to the utility was batched (copy or link-edit utility only), SMP/E debatches the

input and retries the utility separately for each member. For information about this retry processing, see "Recovering After Errors from Utility Processing" on page 64.

5. **Allocate any new libraries required.** Determine where they are to be allocated and then allocate them. Remember that the program directory ordinarily shows how much space will be used—not how much space to allocate for the libraries. Libraries should be allocated with more space than required to allow for later modifications. Usually, twice the required space is recommended to allow for the replacement of every element in the library without running out of space.

   Remember to add the appropriate DDDEF entries to the target zones and DLIB zones into which you will be installing this function.
6. **Check the SMP/E data sets to make sure they have enough space.** If necessary, compress or expand the partitioned data sets. A data set that is easily overlooked in this process is the SMPSTS, which fills very rapidly when you are receiving source updates (JES2 and JES3, for example). Reorganize or expand (if necessary) the CSI data set (using access method services EXPORT and IMPORT).
7. **Create a backup for the volumes affected.** This is a very important step that should not be overlooked. Without a current backup copy, a serious system failure during installation means not only redoing the installation in process but also means going to the last backup level and redoing all the work done since then.
8. **Estimate the time required for APPLY and ACCEPT processing.** Make sure enough time is available to allow these jobs to run to completion.

   The program directory or installation guide may contain information to help you estimate the time required.

Staging the SYSMODs: The RECEIVE Process

When you get a new function as part of a CBPDO, you get one logical tape that contains the function and the unintegrated service. If there is no preventive service for the function, it is not included in your order.

The first step in installing the new function is the RECEIVE process, which reads in the SYSMODs so they can be installed later:

- If you have access to the SMP/E dialogs, you can use the RECEIVE dialog to receive the function and any related service and HOLDDATA.
- You can use the RIMLIB job included on the CBPDO tape to receive the function, service, and HOLDDATA shipped on the CBPDO. For more information, see *z/OS and z/OS.e Planning for Installation* and the documentation that was included with the CBPDO.

Receiving the Function SYSMOD

Function SYSMODs obtained from IBM are packaged in RELFILE format. Before any actual processing takes place, SMP/E must first determine if the SYSMODs packaged in RELFILE format are to be received from tape or DASD. If the SMPPTFIN data set is located on tape, SMP/E assumes that the RELFILEs are on tape. If the SMPPTFIN data set is located on DASD, SMP/E assumes the RELFILEs are on DASD and are cataloged.

During RECEIVE processing, the contents of the RELFILEs are placed into the SMPTLIBs, which are used as temporary storage. SMPTLIBs that are uncataloged

are automatically cataloged by SMP/E. When the RELFILEs are on DASD, SMP/E checks to ensure that the RELFILE and SMPTLIB names are not the same. If they are, RECEIVE processing stops.

Note: Do not delete the SMPTLIBs after the RECEIVE step; they must be retained until after the function is applied and accepted.

Updating the Target Libraries: The APPLY Process

After preparing your target and distribution libraries and receiving the function and any related service and HOLDDATA, the next step is to update your target libraries. Review the program directory for the products you are installing.

When installing a new function, you are concerned with three groups of SYSMODs:

- The function itself
- All PTFs applicable to the function
- All PTFs applicable to other functions that are specified as requisites of the function or service applicable to the function

You may be able to apply all the required SYSMODs with one APPLY command. This method has several advantages:

- It eliminates the need to run the APPLY command several times in order to install the complete set of SYSMODs required.
- You replace elements in the target libraries less often; therefore, there is less risk of running out of space.
- Because the SMP/E overhead and the number of invocations of the system utilities are reduced, overall processing time is decreased.

Therefore, although SMP/E supports the separate installation of a new function and its service, the common installation method is preferred unless the product program directory for other unique installation requirements directs otherwise. This is the method illustrated in subsequent examples. For more information about the APPLY command operands, see The APPLY Command in *SMP/E Commands*.

When you are updating the target libraries, there are actually three distinct SMP/E jobs to be run:

- **Receive additional HOLDDATA.** Before starting the APPLY, you should contact the IBM Support Center to get additional HOLDDATA from the product PSP file or the CORPE PSP file. Obtaining and receiving additional HOLDDATA is covered under "Processing HOLDDATA from PSP Files" on page 121. The other two steps are covered in more detail in the following sections.
- **Run the APPLY CHECK job.** This is a nonupdating mode of APPLY. Its purpose is to help resolve any problems that may prevent the APPLY from completing processing successfully.
- **Apply the SYSMOD updates.** This installs the new function and service into the target libraries.

Checking the Update: The APPLY CHECK Process

The purpose of this step is to determine:

- Whether any errors will occur while the new function is being applied (except for errors that occur as a direct result of an update, such as a target library running out of space). This includes missing DDDEF entries.
- Whether any requisite SYSMODs are missing.

- The target libraries that will be updated.
- The SYSMODs, if any, that will be regressed.

Use the SMP/E dialogs or the following sample job to do an APPLY CHECK for the function and related SYSMODs:

```
//APPLY    JOB 'accounting info',MSGLEVEL=(1,1)
//APPLYCHK EXEC SMPPROC
//SMPCNTL  DD *
SET      BDY(ZOSTGT1)       /* Set to target zone.      */.
APPLY    FORFMID(HXX1200)   /* Apply for this FMID      */
         FUNCTIONS          /*  the function itself     */
         PTFS               /*  and all its PTFs.       */
         GROUPEXTEND        /* Also all requisite PTFS. */
         CHECK              /* But do not update libs.  */
         BYPASS(HOLDSYSTEM  /* Bypass options           */
                HOLDCLASS(UCLREL,ERREL)
               )
         .
/*
```

Researching the APPLY CHECK Reports: As a result of running the APPLY CHECK job, SMP/E produces various messages and reports that you should now use to do further research. Here are some of the errors that may have been detected:

- Some DD statements may be missing. Check the program directory or the *SMP/E Reference* manual to determine why they are required and how they should be specified.
- Some APAR fixes or USERMODs may be regressed. If so, you must determine why. For APAR fixes, you have to get the version of the APAR fix applicable to the new product. For USERMODs, you have to rework the modification to make it applicable to the new function, or eliminate the modification if the product being installed provides the same function. When doing the actual APPLY operation, you may need to specify the BYPASS operand to inform SMP/E that you have resolved these problems.
- Some prerequisite or requisite PTFs may be missing. If so, you should determine whether they can be obtained. Some may already be on an ESO tape you have in-house but have not received; others may not have been shipped, in which case you have to get an early copy of them by contacting the IBM Support Center. Although you can also avoid these conditions by using the BYPASS operand, you are advised not to do this because the regressions have not been resolved.

  Note: Obtaining a product in a CBPDO greatly reduces the amount of work needed to find requisite PTFs, because CBPDOs include all the service for products applicable to the selected SREL.
- Some elements may not have been selected for installation. For each such element, if the current functional owner (that is, FMID) is an IBM product, there may not be a problem; this condition is common and occurs because there are multiple functions with common elements. Check the program directory or installation guide for the product you are installing to determine whether this condition is normal or if it indicates a problem.

  If the FMID is not one for an IBM product, further research is necessary. Contact the current owner of the element to determine how that product is related to the one you are installing.
- Some of the PTFs may not have been selected for installation because of exception SYSMOD conditions identified by the ++HOLD MCSs. When

installing a new function, you may want to research these PTFs further. You can use the reason ID and the comments specified in the ++HOLD MCS to determine which of the following actions is most appropriate:
- Bypass the condition using the BYPASS(HOLDERR) operand
- Do not install the PTF
- Obtain a fix for the APAR

Getting Additional SYSMODs: After doing the research step, you may decide that additional SYSMODs are needed. If so:

1. Obtain the additional SYSMODs by using CBPDO, ESO, IBMLINK, or the IBM Support Center.
2. Receive the additional SYSMODs, using the same source ID value as used when processing the CBPDO tape.
3. Rerun the APPLY CHECK job.

Repeat this process until no new errors are reported.

Updating the Target Library: The APPLY Process

After you have done the APPLY CHECK and the associated research and the other necessary preparation, the APPLY job itself should be fairly simple. The APPLY job does the same checking as the APPLY CHECK and then continues by calling the appropriate system utilities to get all the elements installed.

Note: If a product deletes another product, you cannot use the RESTORE command to back off the applied product and bring back the deleted one.

Use the SMP/E dialogs or the following sample job to apply the function and any related SYSMODs:

```
//APPLY    JOB 'accounting info',MSGLEVEL=(1,1)
//APPLY    EXEC SMPPROC
//SMPCNTL  DD  *
SET      BDY(ZOSTGT1)       /* Set to target zone.      */.
APPLY    FORFMID(HXX1200)   /* Apply for this FMID      */
         FUNCTIONS          /*   the function itself    */
         PTFS               /*   and all its PTFs.      */
         GROUPEXTEND        /* Also all requisite PTFs. */
                            /* No check this time.      */
         BYPASS(HOLDCLASS(UCLREL,ERREL)/*Bypass options.*/
                HOLDSYS(reason-id,...) .
/*
```

If you have obtained additional APAR fixes or USERMODs, you should either specify each of these SYSMODs in the SELECT operand, or if **all** applicable APARs and USERMODs are to be applied, specify the APARS and USERMODS operands.

Note: For most products, you do not have to do any additional processing to get the elements into executable format. However, some products may require you to run additional utilities or to perform extra steps after applying the SYSMODs. See your product documentation for more information.

Testing the New Function

After installing the new function, you should perform two operations:

1. Create a backup of the updated data sets, including any SMP/E data sets affected, in case something happens to the data sets during the next phase.

   Note: If you are doing the installation on a clone of your original system, you already have a backup—your original system.

2. Do some testing before putting the new function into production. This testing should include some of the following:
   - If the product has supplied installation verification procedures (IVPs), they should be run.
   - If your installation has a test job stream, the tests should be run.
   - If the new function could at all affect your ability to IPL the system, try an IPL at this time.

Only after verifying the new function on a noncritical test system should you put it into production. You should not consider the test phase completed until you have run the new function in production mode for some period (as determined by your installation requirements). Only then, if no errors are found, are you ready to go to the next step, updating your distribution libraries.

Updating the Distribution Libraries: The ACCEPT Process

The last major phase of installing a new function is to update the distribution libraries, using the SMP/E ACCEPT command. This is a very critical step: Once the function and its service have been accepted, there is no SMP/E method for removing it from either the target or distribution libraries.

When you are ready to update your distribution libraries, you have the same set of considerations and SMP/E support as described under "Updating the Target Libraries: The APPLY Process" on page 86, and the same three-phase operation:

1. **Receive additional data.** Before starting the ACCEPT process, you should obtain any additional HOLDDATA for the function you are installing by using CBPDO, ESO, IBMLINK, or the IBM Support Center. See "Processing HOLDDATA from PSP Files" on page 121 for additional information.

   Note: If there is a significant time between the APPLY process and ACCEPT process, additional problems may have been reported for which ++HOLD statements have been created. If this new data is not obtained, SMP/E may install PE PTFs into the distribution libraries.
2. **Run the ACCEPT CHECK job.** This is a nonupdating mode of ACCEPT. Its purpose is to help resolve any problems that may prevent the ACCEPT from completing processing successfully.
3. **Accept the SYSMOD update.** This installs the new function and service into the distribution library.

Checking the Update: The ACCEPT CHECK Process

The ACCEPT CHECK job provides the same function for the distribution libraries that the APPLY CHECK job provides for the target libraries. See "Checking the Update: The APPLY CHECK Process" on page 86.

Use the SMP/E dialogs or the following sample job to do an ACCEPT CHECK for the function and related SYSMODs:

```
//ACCEPT   JOB 'accounting info',MSGLEVEL=(1,1)
//ACCEPTCK EXEC SMPPROC
//SMPCNTL  DD *
SET      BDY(DLIB1)         /* Set to DLIB zone.        */.
ACCEPT   FORFMID(HXX1200)   /* Accept for this FMID     */
         FUNCTIONS          /*   the function itself    */
         PTFS               /*   and all its PTFs.      */
         GROUPEXTEND        /* Also all requisite PTFs. */
         CHECK              /* But do not update libs.  */
         BYPASS(HOLDSYSTEM  /* Bypass options           */
```

```
                 HOLDCLASS(UCLREL,ERREL)
                )
        .
/*
```

Researching the ACCEPT CHECK Reports: The ACCEPT CHECK reports should be researched in the same manner as the APPLY CHECK reports (see "Researching the APPLY CHECK Reports" on page 87).

Getting Additional SYSMODs: The process of getting additional SYSMODs or APAR fixes for those PTFs being accepted is the same as during the APPLY process (see "Getting Additional SYSMODs" on page 88).

Updating the Distribution Library: The ACCEPT Process

The ACCEPT process updates the distribution libraries. Use the SMP/E dialogs or the following sample job to accept the function and any related SYSMODs:

```
//ACCEPT   JOB 'accounting info',MSGLEVEL=(1,1)
//ACCEPT   EXEC SMPPROC
//SMPCNTL  DD  *
SET      BDY(DLIB1)         /* Set to DLIB zone.        */.
ACCEPT   FORFMID(HXX1200)   /* Accept for this FMID     */
         FUNCTIONS          /*   the function itself    */
         PTFS               /*   and all its PTFs.      */
         GROUPEXTEND        /* Also all requisite PTFs. */
                            /* No check this time.      */
         BYPASS(HOLDCLASS(UCLREL,ERREL)/*Bypass options */
                HOLDSYS(reason-id,...) .
/*
```

Note: If you have obtained additional APAR fixes or USERMODs, you should either specify each of these SYSMODs in the SELECT operand or, if **all** applicable APARs and USERMODs are to be installed, specify the APARS and USERMODS operands.

Checking Other Zones for Requisites: REPORT CROSSZONE

After installing a function, you may need to check other zones for conditional requisites. A conditional requisite is software (such as service) that must be installed for a given function if another function is also installed. To help automate the research for conditional requisites, the installation logic (SMP/E modification control statements) for a function uses ++IF statements to identify the requisites.

Conditional requisites may be for functions that are installed in different zones. If you have set up automatic cross-zone requisite checking, as described in "Specifying Automatic Cross-Zone Requisite Checking" on page 73, SMP/E will enforce cross-zone requisites during APPLY or ACCEPT processing. Otherwise, you must use the SMP/E REPORT CROSSZONE command after a function and the related service has been installed to manually identify cross-zone requisites defined in the installation logic. To help you install the identified requisites, REPORT CROSSZONE can also write APPLY and ACCEPT commands to the SMPPUNCH data set. So, if you have **not** specified automatic cross-zone requisite checking, and the function you have installed (or any related service) specifies conditional requisites, you should run the REPORT CROSSZONE command against the target and DLIB zones containing that product, as well as against the zones containing the functions identified on the ++IF statements. For more information about the REPORT CROSSZONE command, see Chapter 12, "Identifying Cross-Zone Requisites: The REPORT CROSSZONE Command", on page 137.

Chapter 5. Installing Preventive Service

This chapter describes the steps for installing preventive service. After an introduction to preventive service and a summary of the preventive service process, it discusses the following topics:

- Preparing your system
- Staging the SYSMODs with the RECEIVE command
- Updating the target libraries with the APPLY command
- Testing the new service level
- Updating the distribution libraries with the ACCEPT command

Introduction

You install preventive service through the use of the SMP/E preventive service process. The process uses:

- A tape that contains the preventive service (either a CBPDO tape or an expanded service options (ESO) tape).
- A well-defined set of steps that you should follow to install each service level in order to bring your system up to the current service level.

ESOs are used as input to the software manufacturing database for the service to be included in CBPDOs. As a result, there are many similarities between these two offerings.

If you receive service and HOLDDATA from both CBPDO and ESO tapes, make sure you install them in service level order (for example, service level 0201, followed by service level 0202, and so on). Each PTF that is currently available on a service level has a corresponding source ID. After you have received PTFs from one of these service offerings (ESO or CBPDO), do not try to receive them again from the other.

CBPDO Tapes

CBPDO tapes can be ordered periodically, as you decide to update your system. A CBPDO tape contains the PTFs, HOLDDATA, and PSP upgrade files to bring your system up to a service level that you select. A CBPDO is ordered for a particular feature (CICS, database system, MVS, or NCP). These features correspond to the SRELs to which products are applicable.

You have two options when ordering a CBPDO: you can get products plus service, or service only. (If you are interested just in installing preventive service, order a service-only CBPDO.) With both of these options, you automatically receive service for products for which you are already licensed under a single customer number for a single feature.

The amount of service you receive in your CBPDO depends on your selection of a service level and whether this is your first CBPDO order.

- If you select a service level, you get all service from that level to the current level.
- If you do not select a service level and this is your first CBPDO order, you get all the service shown on the order checklist.

- If you do not select a service level and you have ordered a CBPDO before, you get service following the service level that was shipped in your previous CBPDO.

The CBPDO tape is a standard label (SL) tape, with files arranged as shown in Table 10.

Table 10. Format of a CBPDO Tape

File Number	Processed by SMP/E	Contents
1	No	Documents
2	No	Installation RIMs
3	Yes	HOLDDATA for exception SYSMODs
4	No	Program directories and PSP information
5	Yes	SMPMCS file (MCS statements for SYSMODs on the tape), PTFs, and cover letters
6–*n*	Yes	RELFILEs for products on the tape

ESO Tapes

As an IBM customer, you may periodically receive a customized ESO tape based on your IBM software distribution profile. This tape has the PTFs, HOLDDATA, ++ASSIGN statements, and UCLIN data needed to bring your system up to the current service level.

Service levels are identified according to when they were available correctively. For example:

Service Level	When Available Correctively
0208	August 2002
0301	January 2003
0311	November 2003

The ESO tape is a nonlabeled (NL) tape, with files arranged as shown in Table 11 on page 92.

Table 11. Format of an ESO

File Number	Processed by SMP/E	Contents
1	Yes	++ASSIGN statements PTFs within the requested service level range PTFs that resolve PE PTFs contained in ESO
2	No	Installation and usage instructions
3	No	Softcopy of packing list for all PTFs
4	Yes	HOLDDATA for exception SYSMODs
5	Optional	UCLIN file

Preventive Service Process: Summary

The preventive service process has five phases:
1. Prepare your system.
2. Stage the preventive service.
3. Update the target libraries.
4. Test the system.
5. Update the distribution libraries.

The preventive service phases are the same as those defined in Chapter 6, "Installing Corrective Service", on page 103, although the steps within each phase differ. These phases are the basic SMP/E operations to install any SYSMOD.

Each of these phases consists of a series of steps (SMP/E jobs, research, or invocations of system utilities) that must be done to make sure the preventive service is installed correctly and to ensure the integrity of your system libraries.

The steps defined are the normal steps for the installation of most PTFs. Any PTF that requires special processing will contain instructions for installing it.

Generally, you should first attempt to install all the normal PTFs that you have received and then to install those having special requirements. The intention is to install the maximum number of preventive fixes on your system as soon as possible.

Note: You should let SMP/E manage PE PTFs (PTFs discovered to be in error), rather than researching and resolving them yourself.

The following sections describe, in detail, the steps necessary for each of the preventive service phases.

Note: You can use either the SMP/E dialogs or JCL jobs to receive, apply, and accept preventive service. The basic steps are the same. If you have access to the SMP/E dialogs, you should use them. Otherwise, you can use the steps described in this chapter as examples.

Preparing Your System

Before you start installing preventive service, you should do the following to make sure that your system is ready and that you can recover in case of a serious failure during installation:

- Get the latest HOLDDATA from IBMLINK or the IBM Support Center. You process this HOLDDATA during the next phase.

  As described in Chapter 8, "Managing Exception SYSMODs", on page 113, an ESO may contain PTFs that were discovered to be in error (called PE PTFs) after they were shipped to an IBM software distribution center. You can find out about PE PTFs from IBMLINK or the IBM Support Center and build the corresponding ++HOLD/++RELEASE statements necessary to prevent introducing problems by processing PTFs with known errors.
- Make sure you have the publications you need.
- Estimate the time you need for APPLY and ACCEPT processing. Make sure there is time for these jobs to run to completion.
- Back up your disk packs.

Staging the SYSMODs: The RECEIVE Process

After you prepare your target and distribution libraries, receive the preventive service SYSMODs (PTFs) and the HOLDDATA (including data on the CBPDO or ESO and any obtained from the IBM Support Center) into the SMP/E database (the global zone and the SMPPTS).

- If the service was obtained from a CBPDO, you can use SMP/E dialogs or the RIMLIB job included on the CBPDO tape to receive the service and HOLDDATA shipped on the CBPDO. For more information, see the documentation that was included with your tape.
- If the service was obtained from an ESO, you can use the SMP/E dialogs or the following sample job to receive the service and HOLDDATA.

```
//RECESO EXEC SMPPROC
//SMPHOLD  DD DSN=HOLDDATA,
//            UNIT=TAPE,
//            VOL=(PRIVATE,RETAIN,SER=tape1),
//            LABEL=(4,NL),
//            DCB=(LRECL=80,BLKSIZE=7200,RECFM=FB),
//            DISP=(SHR,PASS)
//SMPPTFIN DD DSN=SMPPTFIN,
//            UNIT=AFF=SMPHOLD,
//            VOL=(PRIVATE,RETAIN,SER=tape1),
//            LABEL=(1,NL),
//            DCB=(LRECL=80,BLKSIZE=7200,RECFM=FB),
//            DISP=(OLD,PASS)
//SMPCNTL  DD *
SET      BDY(GLOBAL)    /* Set to global zone.       */.
RECEIVE  SYSMODS        /* Receive all SYSMODs.      */
         HOLDDATA       /* Receive HOLDDATA.         */
         SOURCEID(      /* Optional: Assign SOURCEID */
           xxxxxxxx)    /* (SOURCEIDs are included   */.
                        /*  on all ESO tapes)       */
```

Note: If multiple ESO service tapes are being RECEIVED, the additional tape volumes must be concatenated under the SMPPTFIN DD card. See File 2 of an ESO tape for further information.

In CBPDO and ESO tapes, PTFs are assigned SOURCEID values by ++ASSIGN statements. You can assign an additional value by specifying it on the RECEIVE command. The SOURCEID operand is used with the MCS operand on the LIST command to list the PTF cover letters. This is used rather than the LIST operand on the RECEIVE command because the output is in a more usable format.

- For ESO tapes, you should substitute a meaningful value for *xxxxxxxx* in each command shown above. The value should be unique and easily tied to an ESO.
- For CBPDO tapes, the recommended format is PDO*yywww*, where *yy* is the year and *ww* the week of the CBPDO tape.

The DCB values shown for SMPHOLD and SMPPTFIN are those used for preventive service when this publication was written. If these values change, use the ones defined for the ESO.

For both CBPDO and ESO tapes, you should call the IBM Support Center to obtain additional HOLDDATA (unless you just received your tape). For additional information, see “Processing HOLDDATA from PSP Files” on page 121. You can use the SMP/E dialogs or the following sample job to process the data set with the HOLDDATA obtained from the IBM Support Center.

```
//RECEIVE   JOB  'accounting info',MSGLEVEL=(1,1)
//RECESM    EXEC  SMPPROC
//SMPHOLD   DD ...data describing your data set
//*           ...must be RECFM=FB, LRECL=80
//SMPCNTL   DD *
SET      BDY(GLOBAL)    /* Set to global zone.      */.
RECEIVE  HOLDDATA       /* Receive HOLDDATA.        */.
/*
```

The HOLDDATA you obtain from the IBM Support Center should be in SMP/E format. If not, or if you are creating your own HOLDDATA, see the *SMP/E Reference* manual to review the syntax for the ++HOLD statements.

Updating the Target Libraries: The APPLY Process

After you prepare your target and distribution libraries and receive all the necessary PTFs and HOLDDATA, update the target libraries. Though most PTFs can be installed directly into the target libraries, some require special processing, such as a fix that must be concurrently installed on all processors in a network.

These PTFs contain a ++HOLD statement that automatically places them into HOLD for SYSTEM action status; that is, SMP/E does not allow them to be installed unless you take some direct action, such as specifying BYPASS(HOLDSYS) on the APPLY command. These PTFs should not be processed immediately; you should attempt to install all PTFs not requiring such actions and then return to process these. For additional information about these PTFs, see "Installing PTFs That Need Special Processing" on page 100.

When installing preventive service, you are concerned with two groups of PTFs:
- All PTFs from the CBPDO or ESO you are installing
- Any other PTFs that are required to install these PTFs

SMP/E provides operands (SOURCEID and GROUP or GROUPEXTEND) on the APPLY command that facilitate the installation of all required PTFs by use of one APPLY command. Installing all PTFs with one APPLY command provides several advantages:
- It eliminates the need to run the APPLY command several times in order to install the complete set of PTFs required.
- It reduces the risk of running out of space, because you are replacing elements in the target libraries less often.
- It decreases overall processing time, because there is less SMP/E overhead and the system utilities are invoked less often.

When you update the target libraries, there are three distinct SMP/E jobs to be run:

1. **Receive additional HOLDDATA.** Before starting the APPLY, you should contact the IBM Support Center to obtain any additional HOLDDATA for the CBPDO or ESO you are installing. This step is required if:
   a. You did not obtain the additional HOLDDATA from the IBM Support Center during the staging phase.
   b. There has been a delay between the RECEIVE and APPLY staging phase and the target update phase.

   We will not discuss this first step further here. If you need to perform this step, see "Staging the SYSMODs: The RECEIVE Process" on page 94.

2. **Run the APPLY CHECK job.** This second step is a nonupdating mode of APPLY, referred to as the *APPLY CHECK run*. Its purpose is to assist in resolving any problems that prevent the APPLY itself from completing processing successfully.
3. **Run the APPLY job.** This third step is the updating mode of APPLY, in which the preventive service is installed into the target libraries.

The following sections describe the last two steps as well as the processing of PTFs that require special processing.

Checking the Update (APPLY CHECK)

The purpose of this step is to determine:

- Whether any errors will occur when you apply a SYSMOD (except for those error conditions that occur as a direct result of an update, such as a target library running out of space)
- Whether any requisite PTFs are missing
- The target system libraries that will be updated during APPLY
- The PTFs or APARs, if any, that will be regressed during APPLY

The GROUP and GROUPEXTEND operands allow SMP/E to include any PTF that may be required to install PTFs on the current service level (PUT0203 in the example that follows). Some of the PTFs on previous tapes may not have been installed, because they were in hold status (PE PTFs) at the time the ESO containing the service level was installed. The current service level may contain fixes for the APARs that caused the original PTFs to be held. These PTFs, because they have module intersections with the PE PTF, must either be prerequisite to the old PTFs or must supersede them so SMP/E can automatically include the old PTFs when the fixing PTF is installed.

The following sample job shows how to do an APPLY CHECK for preventive service:

```
//APPLY    JOB 'accounting info',MSGLEVEL=(1,1)
//APPLYCHK EXEC SMPPROC
//SMPCNTL  DD *
SET      BDY(TGT1)          /* Set to target zone.      */.
APPLY    SOURCEID(PUT0203)  /* Apply this service level */
         GROUPEXTEND        /*  and all requisite PTFs, */
         CHECK              /*  but do not update libs. */
         SELECT(sysmod-id,...) /* Select additional     */
                            /* service if required.     */
         BYPASS(HOLDCLASS(ERREL,UCLREL)
                HOLDSYSTEM) .
/*
```

You may be able to improve SMP/E performance by including the source IDs for previous service levels within the SOURCEID operand. The following job provides an example of an APPLY CHECK job for PTFs in service level 0203:

```
//APPLY    JOB 'accounting info',MSGLEVEL=(1,1)
//APPLYCHK EXEC SMPPROC
//SMPCNTL  DD *
SET      BDY(TGT1)          /* Set to target zone.      */.
APPLY    SOURCEID(PUT0203   /* Apply this service level */
                  PUT0202   /*  and back-level tapes    */
                  PUT0201)  /*  back to some reasonable */
                            /*  level.                  */
         GROUPEXTEND        /* And all requisite PTFs.  */
         CHECK              /* But do not update libs.  */
```

```
        SELECT(sysmod-id,...) /* Select additional     */
                              /* service if required.    */
        BYPASS(HOLDCLASS(ERREL,UCLREL)
               HOLDSYSTEM) .
/*
```

Note: This form of the SOURCEID operand can also be used to group service levels initially in one APPLY command.

If you want to install preventive service only on selected functional areas of the system, you can also specify the FORFMID operand on the APPLY command, specifying either specific function identifiers (FMIDs) or the name of one or more FMIDSETs.

Researching the APPLY CHECK Reports

As a result of running the APPLY CHECK job, SMP/E produces various messages and reports you should now use to perform further research. Here are some of the errors that may be detected:

- Some DD statements may be missing. Check the *SMP/E Reference* manual to determine why they are required and how they should be specified.
- Some APARs or USERMODs may be regressed. If so, you must determine why. For APARs, obtain the version of the APAR fix applicable to the service level. For USERMODs, rework the modification to be applicable to the new service level. When performing the actual APPLY operation, you most likely need to specify the BYPASS operand in order to inform SMP/E that you have resolved these problems.
- Some requisite PTFs may be missing. If so, you should determine how they can be obtained. Some may be on service levels you have not received; others may not have been shipped, in which case you have to obtain an early copy of them by contacting the IBM Support Center. Although you can get around these conditions by using the BYPASS operand, you are advised not to do this because the regressions have not been resolved.
- Some of the PTFs are not installed because of exception SYSMOD conditions identified by the ++HOLD statements. You should ignore these PTFs until a fixing PTF is delivered in a subsequent service level.

  Note: Depending on your requirement to install such PTFs, you can use the reason ID and the comments specified in the ++HOLD statement to determine which of the following actions is most appropriate:
  - Bypass the condition using the BYPASS(HOLDERR) operand
  - Do not install the PTF
  - Obtain a fix for the APAR

A common cause of regression is user modification. When USERMODs are applied to your system, the service level information (RMID or UMID) is altered to reflect these additions. The APPLY CHECK may have flagged a SYSMOD as one that would cause regression.

This regression-handling procedure works under the assumption that you have applied, but not yet accepted, a USERMOD. This means that the USERMOD has applied service to the target libraries, but the service in your distribution library is that which the SYSMOD should be applied against.

You can follow these steps for handling regression:

1. Restore the module from the distribution library back into the target system to back off the USERMOD.

2. Apply the SYSMOD in question to the target system in order to keep SMP/E's information about the target system up to date.
3. Accept the SYSMOD into the distribution libraries.

When USERMODs are applied on a system, it is up to you to ensure that they are at the proper level.

If you reapply your USERMOD at this point, remember to exclude it when accepting the preventive service if you want to be able to restore your system to the level assumed by the next preventive service update.

The following steps describe regression handling.

1. **Restore APARs or USERMODs, if necessary.** Use the RESTORE command to remove the APAR or USERMOD from the target libraries. This places into the target library the versions of the elements that currently exist in the distribution library.

   Use the SMP/E dialogs or the following sample job to restore the SYSMODs:

   ```
   //SMPRESTR JOB  'accounting info',MSGLEVEL=(1,1)
   //RESTORE  EXEC SMPPROC
   //SMPCNTL DD *
   SET     BDY(TGT1)        /* Set to target zone.      */.
   RESTORE                  /* Put DLIB data into       */
                            /*    target libraries.     */
           SELECT(AZ00001,  /* Must be SELECT or GROUP, */
                  AZ00002)  /*     NOT by source ID.    */.
   /*
   ```

2. **Repeat the APPLY CHECK.** This gives you an updated status report to determine that all regression conditions have been addressed.
3. **If a USERMOD or APAR is necessary,** compare the PTF just flagged by APPLY CHECK with the APAR or USERMOD that caused the regression. You may need microfiche or a dump of the module. If any changes are needed, follow the steps below. Otherwise, continue with the APPLY step.
   - Do one of the following:
     - For a USERMOD, add the REWORK operand to the ++USERMOD MCS. The REWORK operand allows the updated SYSMOD to be automatically rereceived, as long as it is more recent than the version that has already been received. This takes the place of rejecting the SYSMOD and receiving it.
     - For an APAR or USERMOD, reject the prior copy from the SMPPTS.

       The SMP/E REJECT job removes the USERMOD or APAR from the SMPPTS. This prevents the prior copy from being applied again.

       A sample REJECT job follows:

       ```
       //SMPREJ  JOB 'accounting info',MSGLEVEL=(1,1)
       //REJECT  EXEC SMPPROC
       //SMPCNTL DD *
       SET      BDY(GLOBAL)     /* Set to global zone.   */.
       REJECT                   /* Remove these two      */
                                /*  SYSMODS from the     */
                S(AZ00001,      /*  PTS and global zone. */
                  AZ00002)      /*                       */.
       /*
       ```

   - **Receive the USERMODs or APARs.** This loads the SYSMODs into the SMPPTS.

Getting Additional SYSMODs

After doing the research step, you may decide that additional SYSMODs are needed. These should be obtained from the IBM Support Center and then received into the SMPPTS.

At this time, you should modify the APPLY command to add a SELECT operand specifying each of the PTFs obtained from the IBM Support Center. An alternative is to assign all such PTFs the same source ID value as the service level, or to assign them a unique value and then add that value to the SOURCEID operand.

This process should continue until no new errors are reported.

Updating the Target Library (APPLY)

If the suggested preparation and all phases of the APPLY CHECK are completed, the APPLY job itself should be fairly simple. The APPLY job performs the same checking as the APPLY CHECK and then calls the appropriate system utilities to install all the elements.

Use the SMP/E dialogs or the following sample job to apply the preventive service:

```
//APPLY    JOB 'accounting info',MSGLEVEL=(1,1)
//APPLY    EXEC SMPPROC
//SMPCNTL  DD  *
SET      BDY(TGT1)          /* Set to target zone.      */.
APPLY    SOURCEID(PUT0203)  /* Apply this service level */
         GROUPEXTEND        /*  and all requisite PTFs, */
         SELECT(UZ00001     /* Plus two other PTFs.     */
                UZ00002     /*                          */
                AZ12345     /* Plus two APAR fixes.     */
                AZ12346)    /*                          */
                            /*  No check this time.     */
         BYPASS(HOLDCLASS(ERREL,UCLREL)
                HOLDSYSTEM) .
/*
```

- If you have obtained additional APAR fixes or USERMODs, you should either specify each of these SYSMODs in the SELECT operand or, if **all** applicable APARs and USERMODs are to be installed, specify the APARS and USERMODS operands.
- If any of the SYSMODs specified in the SELECT list have already been applied and you want to reinstall them, you must also specify the REDO operand on the APPLY command.
- If you want to install preventive service only on selected functional areas of the system, you can also specify the FORFMID operand on the APPLY command, specifying either specific function identifiers (FMIDs) or the name of one or more FMIDSETs.

Installing PTFs That Need Special Processing

There are so many reasons a PTF may require special processing that it is impossible to document how you should handle each case. Any PTF requiring special processing should contain a ++HOLD statement (after all the ++VER statements and before the first element MCS). That ++HOLD statement should be as follows:

```
++HOLD(sysmod-id)           /* Originating SYSMOD ID.   */
      SYSTEM                /* Special processing info. */
      FMID(sysmod-id)       /* Functional owner.        */
      REASON(reason-id)
      COMMENT(....
              ....any amount of comment text
              ....
             )
      .
```

See *SMP/E Reference* for a detailed description of the ++HOLD statement syntax. The comment text within the ++HOLD statement, or in the PTF cover letter, contains a description of all the special processing necessary to install this PTF.

Testing the New Service Level

After having installed the new service, you should perform two operations:

1. Create a backup of the updated data sets, including the SMP/E data sets affected. This ensures that if something happens to the data sets during the next phase, you do not have to repeat all the processing done in previous steps.
2. Perform some testing before putting the service into production. This testing should include some of the following:
   - Run selected product IVP jobs.
   - Run test job streams, if your installation has them.
   - Attempt an IPL.

Only after verifying the service on a noncritical test system should you put that service into production. The test phase should not be considered complete until you have run the service in production mode for some period (as determined by the requirements for your installation). If no errors are found, you are ready to proceed to the next step: updating your distribution libraries.

Updating the Distribution Libraries: The ACCEPT Process

The last major phase of installing preventive service is updating the distribution libraries with the SMP/E ACCEPT command. This is a very critical step. Once the service is accepted, there is no SMP/E method to remove it from either the target or distribution libraries.

When you are ready to update your distribution libraries, you have the same set of considerations and SMP/E support as described under "Updating the Target Libraries: The APPLY Process" on page 95, and the same three-phase operation:

1. **Receive additional HOLDDATA.** Before starting the ACCEPT, you should obtain any additional HOLDDATA for the service level you are installing. This step is required if:
   - You did not obtain the additional HOLDDATA from the IBM Support Center during the staging phase.

- There has been a delay between the RECEIVE staging phase and the ACCEPT DLIB update phase.

2.

   Notes:

   a. If there is a significant time between the APPLY and ACCEPT, additional problems may have been reported for which ++HOLD statements have been created. If this new data is not obtained, SMP/E may install PE PTFs into the distribution libraries.

   b. You may want to run the REPORT ERRSYSMODS command to see whether any SYSMODs that are applied are now in error. For more information, see Chapter 13, "Identifying Installed SYSMODs Affected by Error Holds: The REPORT ERRSYSMODS Command", on page 141.

3. **Run the ACCEPT CHECK job.** The second job is a nonupdating mode of ACCEPT, referred to as the *ACCEPT CHECK run*. Its purpose is to help resolve any problems that prevent the ACCEPT itself from successfully completing processing.
4. **Run the ACCEPT update.** The third job is the updating mode of ACCEPT, in which the preventive service is installed into the distribution libraries.

Note: Special processing may be required during the ACCEPT process. PTFs requiring this processing should be handled in the same manner as during the APPLY process.

Checking the Update (ACCEPT CHECK)

The ACCEPT CHECK job provides the same function for the distribution libraries that the APPLY CHECK job provided for the target libraries. See "Checking the Update (APPLY CHECK)" on page 96.

Use the SMP/E dialogs or the following sample job to do an ACCEPT CHECK for preventive service. This example is an ACCEPT CHECK job for PTFs in service level 0203:

```
//ACCEPT   JOB 'accounting info',MSGLEVEL=(1,1)
//ACCEPTCK EXEC SMPPROC
//SMPCNTL  DD *
SET      BDY(DLIB1)         /* Set to DLIB zone.        */.
ACCEPT   SOURCEID(PUT0203)  /* Accept this service level*/
         GROUPEXTEND(       /*  Include requisite PTFs  */
                NOAPARS     /*  Don't include APARs or  */
                NOUSERMODS) /*  USERMODs                */
         CHECK              /*  but do not update libs. */
         SELECT(sysmod-id,...) /* Select additional     */
                            /* service if required.     */
         BYPASS(HOLDCLASS(ERREL,UCLREL)
                HOLDSYSTEM) .
/*
```

Note: This example can be used for PTFs from either a CBPDO or an ESO.

If you want to install preventive service only on selected functional areas of the system, you can also specify the FORFMID operand on the ACCEPT command, specifying either specific function identifiers (FMIDs) or the name of one or more FMIDSETs.

Researching the ACCEPT CHECK Reports

The ACCEPT CHECK reports should be researched in the same manner as the APPLY CHECK reports (see "Researching the APPLY CHECK Reports" on page 97).

Getting Additional SYSMODs

The procedure for getting additional SYSMODs or APAR fixes from those PTFs being accepted is the same as that followed during the APPLY process (see "Getting Additional SYSMODs" on page 99).

If you obtain additional SYSMODs during the ACCEPT phase, you should process these through the APPLY phase before accepting them.

Updating the Distribution Library (ACCEPT)

The ACCEPT command causes SMP/E to update the distribution libraries. You can use the SMP/E dialogs or the following sample job to accept the preventive service:

```
//ACCEPT   JOB 'accounting info',MSGLEVEL=(1,1)
//ACCEPT   EXEC SMPPROC
//SMPCNTL  DD  *
SET      BDY(DLIB1)         /* Set to DLIB zone.        */.
ACCEPT   SOURCEID(PUT0203)  /* Accept this service level*/
         GROUPEXTEND(       /*  Include requisite PTFs  */
                NOAPARS     /*  Don't include APARs or  */
                NOUSERMODS) /*  USERMODs                */
                            /*  No check this time.     */
         SELECT(sysmod-id,...) /* Select additional     */
                            /* service if required.     */
         BYPASS(HOLDCLASS(ERREL,UCLREL)
                HOLDSYSTEM) .
/*
```

Notes:

1. If you have obtained additional APAR fixes or USERMODs, you should either specify each of these SYSMODs in the SELECT operand or, if **all** applicable APARs and USERMODs are to be installed, specify the APARS and USERMODS operands.
2. If you want to install preventive service only on selected functional areas of the system, you can also specify the FORFMID operand on the ACCEPT command, specifying either specific function identifiers (FMIDs) or the name of one or more FMIDSETs.

Installing PTFs That Need Special Processing

During the ACCEPT process, the considerations for special processing are the same as for the APPLY process (see "Installing PTFs That Need Special Processing" on page 100).

Chapter 6. Installing Corrective Service

This chapter describes the steps for installing corrective service. It discusses these topics:
- An introduction to corrective service
- Building and checking a corrective service fix
- Preparing your system
- Staging the SYSMODs with the RECEIVE command
- Updating the target libraries with the APPLY command
- Testing the corrective service
- Updating the distribution libraries with the ACCEPT command

Introduction

Corrective service is the process of installing a SYSMOD to resolve a specific problem in your system. The problem has usually been brought to your attention because the system has not functioned as expected (for example, an abend has occurred, or jobs are not running as expected).

The first task is to investigate the problem, so that the failing component and module can be identified. *SMP/E Messages, Codes, and Diagnosis* provides helpful information on diagnosing and handling SMP/E problems. This can be done in conjunction with the IBM Support Center. SMP/E can help you work with the IBM Support Center to isolate and obtain a fix for the problem. Useful information includes:
- The function and service level of the module involved
- The service level of your system—that is, the specific SYSMODs that have been installed
- Any USERMODs involved
- The affected load modules

After determining the cause of the error, you want a fix for the problem. There are several possibilities:
1. The problem has already been reported, and a PTF has been created to fix the module. That PTF may not have been shipped on an ESO. If it has been shipped, you should have it in your shop (already received). If not, the IBM Support Center can help you get a copy of it.
2. The PTF identified by the Support Center may have been received but not yet applied. Use the LIST command or the SMP/E Query dialog to check the status of the PTF.
3. The problem has been previously reported. No PTF has been created, but an APAR fix is available either to fix or to bypass the problem.
4. The problem is a new one, and no fix is available. In this case, you have to work with the IBM Support Center to construct a fix for your system.

No matter where you obtain the fix, the installation of that fix is said to be in corrective mode.

Note: You can use either the SMP/E dialogs or JCL jobs to build, receive, apply, and accept corrective service. The basic steps to follow are the same. If you

have access to the SMP/E dialogs, you should use them. Otherwise, you can use the steps described in this chapter as examples.

Building or Checking the Fix

If the fix is a PTF, either from a CBPDO, an ESO or the IBM Support Center, you can assume that the construction of that fix is accurate and the material in this section is not applicable.

If the fix obtained is not a PTF, you should make sure it was constructed accurately. This is true even if the fix obtained from the IBM Support Center is already in SMP/E format (that is, you received a ++APAR type SYSMOD).

If you have to build the fix yourself, see *z/OS Packaging Rules* for rules for constructing APAR SYSMODs and *SMP/E Reference* for details on MCS syntax. (To get started, you might find Chapter 16, "Building a User Modification", on page 147 helpful.) The general format of all ++APAR fixes is:

```
++APAR(xxxxxxx)             /* APAR identifier         */.
++VER (srel)                /* System identifier       */
      FMID(aaaaaaa)         /* Functional area         */
      PRE (                 /* PRE some SYSMODs.       */
           bbbbbbb          /* PRE RMID of element     */
           ccccccc cccccccc /*  and any UMIDs present. */
           ccccccc cccccccc /*                         */
          )                 /*                         */
      SUP (                 /* SUP some SYSMODs.       */
           ddddddd dddddddd /* Fixes incorporated into */
           ddddddd dddddddd /*  this fix               */
          )                 /*                         */.
++ZAP (modname)             /*                         */
      DISTLIB(eeeeeeee)     /* DLIB name               */.
... Some superzap cards here
   or
++MACUPD (macname)          /*                         */
      DISTLIB(eeeeeeee)     /* DLIB name               */.
... Some IEBUPDTE cards here
   or
++SRCUPD (srcname)          /*                         */
      DISTLIB(eeeeeeee)     /* DLIB name               */.
... Some IEBUPDTE cards here
```

You should perform the following checks to ensure the accuracy of the fix:

1. Make sure the value *xxxxxxx* in the ++APAR statement is equal to the APAR number associated with the problem you are fixing.
2. Make sure the system release value (SREL) in the ++VER statement matches one of those defined as an SREL subentry in the TARGETZONE entry for your target zone.
3. Make sure the FMID value *aaaaaaa* in the ++VER statement matches the FMID value in the appropriate element entry in your target zone. You can determine this value by listing the appropriate entry.
4. If the element entry in your target zone has an RMID value different from its FMID value, make sure it is a prerequisite of the APAR fix (that is, make sure the *bbbbbbb* value is accurate). If the RMID and FMID values are equal, the *bbbbbbb* value need not be specified.
5. If the element entry in your target zone has any UMID values, you should first make sure the fix itself was constructed so that it works correctly in that environment.

You should then make sure each of the UMID values is specified in the PRE operand in place of the *ccccccc* values shown in the example. This is not absolutely required, but if it is not done, SMP/E issues warning messages during installation indicating that these SYSMODs may have an intersection with the one you are installing, and therefore may be regressed. Putting the UMID values in the PRE list suppresses these messages.

6. If this SYSMOD is to fix multiple problems, each of the additional APARs that are being fixed should be specified in the SUP operand (*ddddddd* values in the example).
7. Make sure the name in the ++ZAP, ++MACUPD, or ++SRCUPD statement is correct.
8. Make sure the value *eeeeeeee* specified in the DISTLIB operand matches the DISTLIB value in the target zone entry.

   Note: The DISTLIB value is optional, but it is a good idea to specify it to make sure there is no mistake about which element you are dealing with.

Once you have made sure the SYSMOD is accurate, you are ready to start the actual installation process.

Preparing Your System

Corrective service is very different from the installation of a new function or preventive service.

- It usually affects a very limited area of the system.
- It is usually done because a severe problem is affecting the system.
- There is a need for an **immediate** fix.
- The fix usually takes little time to install.

Thus, there is usually no need or time to prepare the system by backing up packs, compressing libraries, and so on. If possible, it is a good idea to have a backup system available in case a problem does occur.

Staging the SYSMODs: The RECEIVE Process

After verifying that the corrective SYSMOD is syntactically correct and specifies the proper set of functions and PTFs, receive that SYSMOD (APAR or PTF) into the SMP/E database so you can install it into the target libraries.

Because corrective service requires a very small number of SYSMODs—often only one—the job of receiving it is very simple. You can use the SMP/E dialogs or the following sample job:

```
//RECEIVE  JOB 'accounting info',MSGLEVEL=(1,1)
//RECEIVE  EXEC SMPPROC
//SMPPTFIN DD ...points to input with SYSMOD
//*             If you put the SYSMOD in a data set
//*                refer to that data set.
//*             If the SYSMOD is in card format
//*                use "DD *" followed by the cards.
//SMPCNTL  DD *
SET      BDY(GLOBAL)        /* Set to global zone.      */.
RECEIVE  SELECT(            /* Receive selected SYSMODs.*/
                xxxxxxx     /* Specify SYSMOD number.   */
               )            /*                          */
         SYSMODS            /* Only process SMPPTFIN -
                               do not look at SMPHOLD.  */.
/*
```

Note: No source ID was assigned. This is because the SYSMOD is installed selectively in the APPLY step. If you want to assign a common value or tag the SYSMOD with some sort of identifier (such as programmer initials), you can use the SOURCEID operand.

If the input data set contains only the SYSMODs you are installing for this corrective service problem, you can omit the SELECT operand. SMP/E then attempts to process all the SYSMODs in the SMPPTFIN input data set.

Updating the Target Libraries: The APPLY Process

After receiving the corrective service, you are ready to install it into the target libraries. You should not attempt to install the SYSMODs without first verifying them. If you have done all the proper checking before this time, the SYSMODs should be installed correctly. However, if you have overlooked something, the direct installation may cause unexpected results.

Checking the Update (APPLY CHECK)

The purpose of this job is to verify that the SYSMOD(s) can be installed correctly and that you understand what libraries and load modules in the system are affected. You can use the SMP/E dialogs or the following sample job to do an APPLY CHECK for corrective service:

```
//APPLY    JOB 'accounting info',MSGLEVEL=(1,1)
//APPLYCHK EXEC SMPPROC
//SMPCNTL  DD *
SET      BDY(TGT1)          /* Set to target zone.      */.
APPLY    SELECT(            /* Install selected SYSMOD. */
               xxxxxxx      /* Specify SYSMOD name here.*/
              )             /*                          */
         CHECK              /* In check mode only.      */.
```

Researching the APPLY CHECK Reports

Review the reports from the check, looking for the following types of information:

- Were any error messages produced? If so, determine the cause and fix the problem.
- Will any SYSMODs be regressed? If so, determine how to resolve the problems.
- Are any other areas of the system affected?

Using HOLDDATA to Assist in Identifying Fixes: If the SYSMOD you are installing is a PTF (obtained from a CBPDO, an ESO, or directly from the IBM Support Center), SMP/E may have some HOLDDATA stored applying to that PTF. If so, the reports will indicate all the reason IDs preventing PTF installation. You should use these reason IDs to determine what the errors are. This can be done by:

1. Listing the SYSMOD and MCS entries for the PTF.
2. Looking at the ++HOLD statements that are listed.
3. Using the COMMENT field in the ++HOLD statement to determine the cause of the error. If the COMMENT field is not present or does not describe the problem adequately, ask the IBM Support Center for further information.
4. Determining whether the error in the PTF is critical enough to stop it from being installed. Remember that you are trying to fix an existing problem; you may decide to install the PTF to fix that problem because the exposure is minimal.

5. If necessary, contact the IBM Support Center to obtain a corrective fix for the PTF. If, in the preceding step, you decided that the PTF should be installed immediately to fix your problem, you should perform this step at some later date.

Getting Additional SYSMODs

If the research of the APPLY CHECK reports discloses that additional SYSMODs are required, these should be obtained in the same manner as the original corrective SYSMOD.

Updating the Target Library (APPLY)

Once the APPLY CHECK has run to your satisfaction, you are ready to install the fix using the SMP/E dialogs or the following sample job to apply the corrective service:

```
//APPLY    JOB 'accounting info',MSGLEVEL=(1,1)
//APPLY    EXEC SMPPROC
//SMPCNTL  DD *
SET      BDY(TGT1)          /* Set to target zone.      */.
APPLY    SELECT(            /* Install selected SYSMOD. */
               xxxxxxx      /* Specify SYSMOD name here.*/
              )             /*                          */
                            /* Note no check operand.   */.
```

Testing the Corrective Service

The testing needed after you install a corrective fix depends on the type of problem you encountered. It may range from no testing to running the job in which the error was detected.

Updating the Distribution Libraries: The ACCEPT Process

Once you have installed the corrective service into the target libraries, you must decide whether you want to update the distribution libraries. You should base this decision on the products involved and on your processing requirements.

The following is a consideration for **not** accepting corrective service:

Corrective service has not been tested and, therefore, may be found to be in error at some later date. Once you have accepted the SYSMODs, there is no RESTORE capability.

The following are some of the considerations for **accepting** corrective service:

- If you do not accept the SYSMOD and you perform a system generation, all that service is lost and must be reinstalled.
- If you must restore a SYSMOD, the work required increases with the number of SYSMODs that have been applied but not accepted. All intersecting SYSMODs must be restored, and then all but the desired SYSMOD must be reapplied. This is especially true for source-modified products.

The following sections describe the steps you should use, assuming you have decided to accept some corrective service.

Checking the Update (ACCEPT CHECK)

The ACCEPT CHECK job provides the same function for the distribution libraries that the APPLY CHECK job provided for the target libraries. It is important because the function and service level of the modules in the distribution libraries

may be different from that in the target libraries. You can use the SMP/E dialogs or the following sample job to do an ACCEPT CHECK for corrective service:

```
//ACCEPT   JOB 'accounting info',MSGLEVEL=(1,1)
//ACCEPTCK EXEC SMPPROC
//SMPCNTL  DD *
SET      BDY(DLIB1)         /* Set to DLIB zone.        */.
ACCEPT   SELECT(            /* Install selected SYSMOD. */
               xxxxxxx      /* Specify SYSMOD name here.*/
              )             /*                          */
         CHECK              /* In check mode only.      */.
```

Researching the ACCEPT CHECK Reports

You should research the ACCEPT CHECK reports in the same way as for the APPLY process (see "Researching the APPLY CHECK Reports" on page 106).

Using HOLDDATA to Assist in Identifying Fixes: If SMP/E reported any exception SYSMOD data during the APPLY CHECK process, you should expect to see the same information during the ACCEPT CHECK process. If you have processed any HOLDDATA between the APPLY and ACCEPT, additional information may be reported. This information should be handled in the same manner as the APPLY information.

Getting Additional SYSMODs

If additional SYSMODs are required to ACCEPT the corrective service, you should obtain them in the same manner as the original corrective service SYSMOD.

Note: If you obtain additional SYSMODs, make sure you process them through the APPLY and test phases before accepting them.

Updating the Distribution Library (ACCEPT)

Once the ACCEPT CHECK runs to your satisfaction, you are ready to accept the fix. Use the SMP/E dialogs or the following sample job to accept the corrective service:

```
//ACCEPT   JOB 'accounting info',MSGLEVEL=(1,1)
//ACCEPT   EXEC SMPPROC
//SMPCNTL  DD *
SET      BDY(DLIB1)         /* Set to DLIB zone.        */.
ACCEPT   SELECT(            /* Install selected SYSMOD. */
               xxxxxxx      /* Specify SYSMOD name here.*/
              )             /*                          */
                            /* Note no check operand.   */.
```

Chapter 7. Installing a User Modification

This chapter describes the steps for installing a user modification (USERMOD). After an introduction to USERMODs, it describes the following processes:

- Preparing your system
- Staging the USERMOD with the RECEIVE command
- Updating the target libraries with the APPLY command
- Testing the USERMOD
- Updating the distribution libraries with the ACCEPT command

Introduction

A USERMOD is a SYSMOD used to make a modification to some IBM-supplied software element (module, macro, source, or data element) to implement a new function or to provide a hook into a user program that provides that function.

A USERMOD should not be confused with an APAR SYSMOD (corrective fix), even if you built the initial version of that fix to fix a problem immediately. For a description of how to construct USERMODs, see Chapter 16, "Building a User Modification", on page 147. For details on the syntax of the MCS statements used in constructing USERMODs, see the *SMP/E Reference* manual. The *z/OS Packaging Rules* contains additional information on when a USERMOD should be used. The rest of this chapter assumes that you have properly constructed the USERMOD and are now ready to install it.

Note: You can use either the SMP/E dialogs or JCL jobs to receive, apply, and accept USERMODs. The basic steps to follow are the same. If you have access to the SMP/E dialogs, you should use them. Otherwise, you can use the steps described in this chapter as examples.

Preparing Your System

You must determine the amount of system preparation necessary for your USERMOD. If it is extensive and affects critical components of the system, you should perform the same tasks as defined under "Preparing Your System" on page 84 or "Preparing Your System" on page 93. If it is a minor change, affecting very few modules and not critical to the operation of the system, no preparation is needed.

Staging the SYSMODs: The RECEIVE Process

Because a USERMOD is generally processed as a single SYSMOD, processing is very similar to that for corrective service; that is, it is received by use of the SELECT option. Use the SMP/E dialogs or the following sample job to receive the USERMOD:

```
//RECEIVE  JOB 'accounting info',MSGLEVEL=(1,1)
//RECEIVE  EXEC SMPPROC
//SMPPTFIN DD ...points to input with your USERMOD
//*              If you put the USERMOD in a data set
//*                 refer to that data set.
//*              If the USERMOD is in card format
//*                 use "DD *" followed by the cards.
//*              Create your data set in LRECL=80,
//*                 FB format.
```

```
//SMPCNTL  DD *
SET      BDY(GLOBAL)        /* Set to global zone.      */.
RECEIVE  SELECT(            /* Receive selected SYSMODs.*/
               xxxxxxx      /* Specify USERMOD number.  */
              )             /*                          */
         SYSMODS            /* Only process SMPPTFIN -
                               do not look at SMPHOLD.  */.
/*
```

Note: No source ID was assigned, because the SYSMOD is installed selectively in the APPLY step. If you want to assign a common value to all the USERMODs or tag each of them with some sort of identifier (such as programmer initials), you can use the SOURCEID operand.

If the input data set contains only USERMODs that you want to receive now, you can omit the SELECT operand. SMP/E then attempts to process all SYSMODs in the SMPPTFIN input data set.

Updating the Target Libraries: The APPLY Process

After receiving the USERMOD, you are ready to install it into the target libraries. You may be tempted to install the SYSMODs without first performing the verification pass. If you have constructed your USERMOD correctly, it should install correctly. However, if you have overlooked something, the direct installation may cause unexpected results. Thus, it is advisable to perform the verification pass.

Checking the Update (APPLY CHECK)

The purpose of this job is to verify that the SYSMODs are installed correctly and that you understand which libraries and load modules in the system are affected. Use the SMP/E dialogs or the following sample job to do an APPLY CHECK for the USERMOD:

```
//APPLY    JOB 'accounting info',MSGLEVEL=(1,1)
//APPLYCHK EXEC SMPPROC
//SMPCNTL  DD *
SET      BDY(TGT1)          /* Set to target zone.      */.
APPLY    SELECT(            /* Install selected SYSMOD. */
               xxxxxxx      /* Specify SYSMOD name here.*/
              )             /*                          */
         CHECK              /* In check mode only.      */.
```

Note: At times, it may be necessary to reinstall a USERMOD—for example, after the installation of a PTF. If you are reinstalling it, the APPLY REDO operand is necessary. You may also have to specify one of the BYPASS operands; that depends on the relationship between your USERMOD and the PTF that was installed.

Researching the APPLY CHECK Reports

Review the reports from the APPLY CHECK process, looking at the following types of information:

- Were any error messages produced? If so, determine the cause and fix the problem.

  A common error here is that the FMID specified on the ++VER modification control statement did not match the FMID value in the element entry; therefore, SMP/E does not select the element to be installed. This condition does not stop the USERMOD from being installed. However, messages are issued to say which elements were not selected.

- Will any SYSMODs be regressed? If so, determine how to resolve the problems.

Updating the Target Library (APPLY)

Once the APPLY CHECK runs to your satisfaction, you are ready to install the USERMOD. Use the SMP/E dialogs or the following sample job to apply it:

```
//APPLY    JOB 'accounting info',MSGLEVEL=(1,1)
//APPLY    EXEC SMPPROC
//SMPCNTL  DD *
SET      BDY(TGT1)          /* Set to target zone.      */.
APPLY    SELECT(            /* Install selected SYSMOD. */
               xxxxxxx      /* Specify SYSMOD name here.*/
              )             /*                          */
                            /* Note no check operand.   */.
```

Testing the USERMOD

The amount of testing needed after the installation of a USERMOD depends on the changes you are making. You may want to review the recommendations found under "Testing the New Function" on page 88 and "Testing the New Service Level" on page 100.

When originally constructing your USERMOD, you may want to provide a document similar to a program directory, containing some of the following information:

- The elements affected.
- The areas within each element.
- Externals of the change.
- An IVP job that can be used to ensure that the USERMOD is working correctly. This can be used after subsequent preventive service is applied or if the USERMOD must be reinstalled because a new release of the IBM product is installed.

This information may be helpful to the next system programmer responsible for installing and maintaining your USERMOD.

Updating the Distribution Libraries: The ACCEPT Process

Once you have installed the USERMOD into the target libraries, you must decide whether you want to update the distribution libraries. While this decision is up to you, IBM generally does **not** recommend the accepting USERMODs, because if a problem is encountered in the modified modules, you may be asked to re-create the problem using an unmodified version. If you have accepted the USERMOD, you cannot create an unmodified version of the module unless you are also maintaining a separate set of distribution libraries without the USERMODs.

Note: You can use ++HOLD statements to prevent USERMODs from being accepted. For each USERMOD that you want to keep from being accepted, follow these steps after applying the USERMOD:

1. Create a ++HOLD statement with a user reason ID that you plan to use only for USERMODs that are not supposed to be accepted. Here is an example:

   ```
   ++HOLD(usermod) USER
   REASON(NOUSERM)
   COMMENT(do not accept this usermod).
   ```

2. Run the SMP/E RECEIVE command to read in the ++HOLD statement. Use the SMPHOLD DD statement to point to the file or data set containing the ++HOLD statement.

Because of the user hold, this USERMOD can be accepted only if you bypass the specific user reason ID. The SYSMOD will not be automatically accepted if you specify USERMOD or the specific SYSMOD ID on the ACCEPT command without bypassing the user reason ID.

Be aware that if you receive the ++HOLD statement before applying the USERMOD, you must bypass the user hold reason ID in order to apply the USERMOD.

Chapter 8. Managing Exception SYSMODs

This chapter explains how SMP/E manages SYSMODs that require special processing. It discusses these topics:
- An introduction to exception SYSMODs
- What SMP/E does with HOLDDATA
- Sources of HOLDDATA
- Steps for processing the data

Introduction

Most SYSMODs you receive from IBM can be installed without additional considerations; you can simply receive, apply, and then accept them. For some SYSMODs, however, this is not possible. Examples of such SYSMODs are:
- SYSMODs that were sent out to correct a problem but that either have not fixed the problem or have introduced a new problem. These are called *PTFs in error*, or *PE PTFs*.
- SYSMODs that require special installation processing, such as a fix that must be concurrently installed on all processors in a network.
- SYSMODs that introduce changes into the system that you should be made aware of, such as changes to operator messages or critical documentation changes.

In SMP/E terms, these SYSMODs are called *exception SYSMODs*. SMP/E supplies a function to automate the management of exception SYSMODs. SMP/E supports three categories of exception SYSMODs:
- **Error.** PTFs in error (PE PTFs).
- **System.** SYSMODs identified by IBM as requiring special processing or notification.
- **User.** Any SYSMODs that you identify as requiring special processing.

Two MCSs are used to manage exception SYSMODs:
- ++HOLD puts a SYSMOD into exception status.
- ++RELEASE removes a PE PTF from exception status when it has been determined that the PTF was held erroneously.

++HOLD statements for system holds are usually built as part of the held PTF. ++RELEASE statements and ++HOLD statements for error or user holds must be in SMPHOLD. ++HOLD and ++RELEASE statements provided by SMPHOLD (external holds) identify the following:
- The SYSMOD ID of the exception SYSMOD (that is, the SYSMOD being held).
- The exception SYSMOD category.
- The FMID to which that ++HOLD applies.
- The reason the SYSMOD is being put into or was in exception status. This is a 1- to 7-character alphanumeric string called the *reason ID*.
  - For **error**-category exception SYSMODs, SMP/E expects the reason ID to be the SYSMOD ID of the APAR reporting the problem.
  - For **system**-category exception SYSMODs, SMP/E expects the reason ID to be a short description of the action required.

– For **user**-category exception SYSMODs, SMP/E makes no assumption about what the reason ID represents.

For more information about reason IDs, see the *SMP/E Reference* manual.

- Text describing why the SYSMOD is being put into exception status. This field is only for ++HOLD statements.
- An alternative way to release the exception SYSMOD. This field is only for ++HOLD statements.

  Every ++HOLD statement specifies a HOLD category of ERROR, SYSTEM, or USER. In addition to one of these categories, a ++HOLD statement may include a HOLD CLASS, which is an alternative way to release a held SYSMOD. For example, an exception SYSMOD may fix a problem more severe than the problem it introduces. The ++HOLD statement for that SYSMOD would have an ERROR reason ID that matches an APAR ID and a CLASS of ERREL.

++HOLD statements provided within a SYSMOD identify the same information. However, even though these internal holds are effective against the containing SYSMOD, the SYSMOD ID specified on the hold may be different from that of the containing SYSMOD, as long as the SYSMOD ID specified on the hold is superseded by the containing SYSMOD.

SMP/E then manages exception SYSMODs by actually managing the resolution of the problems described by the reason ID specified on the ++HOLD statement.

Subsequent sections of this chapter describe how SMP/E uses HOLDDATA during the installation of a SYSMOD, where the exception SYSMOD statements come from, and how to process them. The chapters on the RECEIVE command, the APPLY command, and the ACCEPT command in *SMP/E Commands* contain a much more detailed explanation of the material covered here.

What SMP/E Does with the HOLDDATA

This section describes what SMP/E does with the HOLDDATA when processing the various commands associated with installing and removing SYSMODs.

Note: You must provide SMP/E with the most current HOLDDATA possible to get the most benefit from this support.

Initial Entry into Staging Data Sets: RECEIVE

The RECEIVE command tells SMP/E to take the HOLDDATA from the input data set on which it was delivered and store it in the SMP/E database.

The two operands that control input processes are:

- **SYSMOD**, which tells SMP/E to process the SYSMODs from the data set specified by the SMPPTFIN DD statement
- **HOLDDATA**, which tells SMP/E to process the HOLDDATA (++HOLD and ++RELEASE statements) from the data set or file in the hierarchical file system specified by the SMPHOLD DD statement

You can specify one or both operands on the RECEIVE command. If neither operand is specified, SMP/E attempts to receive the SYSMODs from SMPPTFIN, and HOLDDATA from SMPHOLD.

When receiving a SYSMOD, SMP/E creates two entries:

1. An MCS entry is created on the SMPPTS. This entry is a copy (possibly compacted) of the SYSMOD as it appeared in the SMPPTFIN data set.
2. A SYSMOD entry is created in the global zone. This entry contains information that describes the installation requirements and element content of the SYSMOD.

When receiving the HOLDDATA, SMP/E also creates (or modifies) two entries:

1. A HOLDDATA entry is created (or modified) in the global zone. This entry is an exact copy of the ++HOLD statements as they appeared in SMPHOLD. The name of the entry is the ID of the SYSMOD affected by this ++HOLD statement. The HOLDDATA entry for a single SYSMOD can contain multiple ++HOLD statements.

   Note: When a ++RELEASE statement is processed, SMP/E removes the corresponding ++HOLD statement from the HOLDDATA entry. When all ++HOLD statements are removed, the HOLDDATA entry is automatically deleted.
2. A SYSMOD entry is created (or modified) in the global zone. This entry contains information that describes the exception SYSMOD conditions.

   For each ++HOLD statement processed, SMP/E updates the global zone SYSMOD entry to add a HOLD reason ID subentry. There are three types of HOLD reason ID subentries, HOLDERROR, HOLDSYSTEM, and HOLDUSER, corresponding to the three categories of exception SYSMODs.

   Note: When a ++RELEASE statement is processed, SMP/E removes the corresponding reason ID from the global zone SYSMOD entry. Do not use the ++RELEASE statement to install a SYSMOD with an unresolved reason ID. Use the appropriate BYPASS operand instead.

Updating Target Libraries: APPLY

When SMP/E applies a SYSMOD, SMP/E checks to see if that SYSMOD is currently in exception SYSMOD status by seeing if there are any HOLD reason ID subentries in the global zone SYSMOD entry. If so, SMP/E makes sure each reason ID is resolved before allowing the SYSMOD to be installed.

For an error reason ID to be resolved, **at least one** of the following conditions must be met:

- The reason ID must be superseded by another SYSMOD being installed.
- The reason ID must already exist as a SYSMOD entry in the target zone.
- You must specify BYPASS(HOLDERROR) on the APPLY command to show that you are aware that an unresolved exception SYSMOD is being installed.
- If there is a HOLD CLASS associated with the reason ID, you can specify BYPASS(HOLDCLASS) on APPLY to indicate that you are using an alternative way to resolve the reason ID.

  See the BYPASS operand in The APPLY Command in *SMP/E Commands* for additional information.

Internal holds are considered resolved if any of the following conditions are met:

- The SYSMOD ID specified on the ++HOLD defining the exception is found as a SYSMOD entry in the target zone
- The SYSMOD ID specified on the ++HOLD defining the exception is being superseded by a SYSMOD being applied concurrently

- The applicable BYPASS operand is specified.

You can resolve external system and user reason IDs by specifying BYPASS(HOLDSYSTEM) or BYPASS(HOLDUSER) on the APPLY command. If there is a HOLD CLASS associated with the reason ID, you can specify BYPASS(HOLDCLASS) on APPLY to indicate that you are using an alternative way to resolve the reason ID.

If you choose to resolve a reason ID by using the BYPASS operand, you must do any required processing at the appropriate time. Otherwise, errors related to the undone processing may occur, even though the reason ID was considered resolved.

Note: You may use the ++RELEASE statement for user category reason IDs if you want to unconditionally release the SYSMOD for all systems. Remember that, unlike BYPASS, ++RELEASE actually deletes the ++HOLD statement. If you plan to use the user category ++HOLD statement, see the *SMP/E Reference* manual for more information on the naming conventions for reason IDs.

If all reason IDs are resolved, SMP/E allows the SYSMOD to be applied. If any remain unresolved, SMP/E prevents the SYSMOD, and any other SYSMODs dependent on this one, from being installed.

SMP/E leaves the reason IDs in the global zone SYSMOD entry when the SYSMOD is applied, so if the SYSMOD is applied on another system later, the same checking is done on that system. If the information had been deleted during the first APPLY, SMP/E would not recognize the problem when the SYSMOD is applied to subsequent systems. Therefore, the ++RELEASE statement **should not be used** to install an exception SYSMOD with an unresolved reason ID. The appropriate BYPASS operand should be used instead.

In summary, SMP/E ensures that no known problems are introduced into your system by managing those problems at the level of the individual problem, rather than the SYSMOD level. It is, therefore, very important that SMP/E have the most current information on exception SYSMODs. For more information on the importance of having current HOLDDATA and what you **must do** to provide that information to SMP/E, see "How to Process HOLDDATA" on page 118.

Updating Distribution Libraries: ACCEPT

Exception SYSMOD processing is the same when accepting a SYSMOD as when applying one, except that the appropriate distribution zone is used to determine whether the fixes for the reason IDs have been installed.

Removing HOLDDATA from SMP/E Data Sets

There are various ways to remove HOLDDATA from SMP/E data sets.

After a Successful ACCEPT

When accepting a SYSMOD, if SMP/E determines that the global zone SYSMOD entry and the SMPPTS MCS entry are to be deleted, SMP/E deletes any HOLDDATA associated with those SYSMODs. Once the SYSMOD and MCS entries for a SYSMOD have been deleted, you will probably not install that SYSMOD on any other systems, so you would not need the HOLDDATA again.

During RESTORE Processing

HOLDDATA is never deleted during RESTORE processing. The assumption is that you may later want to reapply the SYSMODs you are restoring.

With the REJECT Command

If you are using the SMPPTS as a history file of all SYSMODs, you may eventually want to purge some of those SYSMODs. To do this, use the REJECT command. You can use the HOLDDATA operand to have SMP/E delete not only the global zone SYSMOD and SMPPTS MCS entries, but also any associated HOLDDATA.

Sources of HOLDDATA

Besides the data you create, these are the main sources of HOLDDATA provided by IBM:

- CBPDO tapes
- Expanded service options (ESO) tapes
- Preventive service planning (PSP) information
- IBM service web pages

This section describes these sources.

CBPDO Tapes

One of the primary means of obtaining HOLDDATA is CBPDO tapes. IBM custom-builds these tapes to provide the products and service you request, taking into consideration whether this is your first CBPDO order.

- If you select a particular service level, you get HOLDDATA for all service from that level to the current level.
- If you do not select a service level and this is your first CBPDO order, you get HOLDDATA for all the service shown on the order checklist.
- If you do not select a service level and you have ordered a CBPDO before, you get HOLDDATA for service following the last service level that was shipped in your previous CBPDO.

The HOLDDATA on a CBPDO tape has been customized to your product set. That is, it contains only data applicable to PTFs for those products **within a given feature** for which you are licensed under a single customer number. However, it does not reflect the contents of any specific system within the establishment defined by that customer number.

The HOLDDATA on the CBPDO tape should be processed immediately on receipt of the tape. You can use either the SMP/E dialogs or the RIMLIB jobs provided with the CBPDO tape to receive the HOLDDATA. For more information, see the documentation that came with the CBPDO tape.

ESO Tapes

Another way to obtain HOLDDATA is through ESO tapes. IBM regularly creates the service levels shipped on these tapes, then custom-builds ESOs for users and makes the tapes available through either subscription orders or special request orders.

The HOLDDATA on an ESO tape has been customized to your product set. That is, it contains only data applicable to PTFs for those products **within a given feature**

for which you are licensed under a single customer number. However, it does not reflect the contents of any specific system within the establishment defined by that customer number.

The HOLDDATA on the ESO should be processed immediately on receipt of the tape. For more information on processing the ESO, see "Processing HOLDDATA from an ESO" on page 120 and "Example of Processing HOLDDATA" on page 121.

PSP Information

Once a service level has been created, there is no further opportunity to change the HOLDDATA on that tape, even though new errors are reported. It is, however, important that you have this error information when you are about to install a CBPDO or an ESO. Otherwise, you may direct SMP/E to install a PE PTF, thus introducing a problem while fixing one. To solve this problem, PSP files have been set up to hold this additional HOLDDATA. Within each SREL, there is one PSP file for each service level.

When a service level is created, a PSP file is also created. As problems (APARs) are reported, appropriate ++HOLD statements are added to the applicable PSP files. IBM determines the applicable PSP files as follows:

1. Determine the PTF that introduced the APAR.
2. If the PTF is not yet assigned a monthly service level, add a ++HOLD statement to the CORPE PSP file as well as the most current monthly bucket.

   Note: The CORPE PSP file is for PE PTFs available correctively, but not yet assigned to a monthly service level. The current monthly bucket will contain ++HOLD statements for corrective service PTFs as well as those assigned to a monthly service level.
3. If the PTF is available on a service level, determine the service level on which that PTF was first shipped.
4. Add a ++HOLD statement to the PSP file corresponding to that service level.
5. Add a ++HOLD statement to each PSP file corresponding to any service levels shipped after the one determined in the previous step, up to the current service level.
6. Add a ++HOLD statement for the next service level.
7. No further PSP files are updated.

Before installing a CBPDO tape, an ESO tape, or PTFs in corrective mode, use IBMLINK or contact the IBM Support Center to get the information in the applicable PSP file. For more information on how to use the PSP information, see "Processing HOLDDATA from PSP Files" on page 121 and "Example of Processing HOLDDATA" on page 121.

How to Process HOLDDATA

The management of exception SYSMODs is a very important part of SMP/E. SMP/E's ability to manage exception SYSMODs, however, is limited by the quality and timeliness of the HOLDDATA made available to it. To gain the full advantage of this function, you must understand how SMP/E expects the three HOLDDATA input sources to be used and the times during which SMP/E expects them to be used.

The following steps summarize the process for managing exception SYSMOD data:

1. Receive all new products as you get them, or use UCLIN to add the FMIDs of the new products to the global zone. This allows you to process exception SYSMODs for them. Then receive any associated HOLDDATA shipped with the product.
2. Receive HOLDDATA from subsequent CBPDO or ESO tapes **in service level order**. Remember to do the following:
   - Receive all new products as you get them so you can process exception SYSMODs for them.
   - Receive HOLDDATA as soon as you get your CBPDO or ESO tapes.
3. Before doing preventive service, do the following:
   a. Get the PSP file associated with the last service level for which you processed HOLDDATA, and receive this additional HOLDDATA for your products.
   b. Get the CORPE PSP file. This contains PE PTFs that are available correctively, but are not yet available in a service level.
   c. List and review HOLDDATA for SYSTEM HOLDs and, if possible, handle the required special conditions. Then apply and accept these processed SYSMODs by specifying BYPASS(HOLDSYS) and listing the individual SYSMOD IDs on the SELECT operand. This helps makes sure all available service is installed when you do preventive service.

Be sure to review "Example of Processing HOLDDATA" on page 121. This example shows why you should follow the procedures defined.

Processing HOLDDATA from a CBPDO Tape

When you receive a CBPDO tape, the HOLDDATA it contains is based on the service level you selected and on whether this is your first CBPDO. This HOLDDATA pertains both to the PTFs actually on that tape and to PTFs shipped on previous tapes. Follow these steps to process the HOLDDATA:

1. Receive the HOLDDATA from the CBPDO tape **as soon as you get the tape**. Use the SMP/E dialogs or the RIMLIB job provided with the CBPDO tape.

   By receiving the HOLDDATA as soon as possible, you make sure SMP/E has the most current information available. Therefore, if you try to install any PTF in response to a problem on your system and that PTF is in error, SMP/E knows this and warns you so you can weigh the effect of installing the known problem against the effect of fixing the problem you have encountered.
2. Receive the SYSMODs from the CBPDO tape **as soon as you get the tape**. Use the SMP/E dialogs or the RIMLIB job provided with the CBPDO tape.

   This makes sure that all available PTFs are ready to be installed. If you find a problem in your system and determine that a PTF must be installed in corrective mode, you have a better chance of having that PTF and all its requisites readily available on the SMPPTS.

Note: You can receive the SYSMODs and HOLDDATA separately or in the same job.

By following these procedures, you are essentially making a trade-off: system resources as increased DASD space for the SMPPTS against the time the system programmer would spend on searching for the service level with the required PTF and on fixing problems caused by installing PE PTFs.

One important part of this procedure is that the HOLDDATA on each CBPDO and ESO **must be received in chronological order**. SMP/E processes the ++HOLD and

++RELEASE statements in the order in which they are encountered. Therefore, there can be an exposure if you receive the data out of sequence. For instance, the tapes may be set up so that one contains a ++HOLD for a PTF and a subsequent one contains a ++RELEASE for the same PTF. If the tapes are processed in the wrong order, the RELEASE statement is processed first, and then the HOLD statement. As a result, the PTF remains held.

Processing HOLDDATA from an ESO

When you receive your expanded service options (ESO) tape, the HOLDDATA it contains pertains both to the PTFs actually on that tape and to PTFs shipped in previous service levels. SMP/E and exception SYSMOD support have been designed to work most efficiently and effectively if you adhere to the following processing guides:

1. Receive the HOLDDATA file of the ESO **as soon as you get the tape** using the SMP/E dialogs or the following sample job:

   ```
   //RECHOLD  JOB 'accounting info',MSGLEVEL=(1,1)
   //RECEIVE  EXEC SMPPROC
   //SMPHOLD  DD ...data describing exception file
   //SMPCNTL  DD *
   SET      BDY(GLOBAL)        /* Set to global zone.   */.
   RECEIVE  HOLDDATA           /* Receive only exception
                                  SYSMOD data.          */.
   /*
   ```

   By receiving the HOLDDATA as soon as possible, you make sure SMP/E has the most current information available. Therefore, if you try to install any PTF in response to a problem on your system and that PTF is in error, SMP/E knows this and warns you so you can balance the effect of installing the known problem against the effect of fixing the problem you have encountered.

2. Receive the SYSMOD file of the ESO **as soon as you get the tape** using the SMP/E dialogs or the following sample job:

   ```
   //RECPTFS  JOB 'accounting info',MSGLEVEL=(1,1)
   //RECEIVE  EXEC SMPPROC
   //SMPPTFIN DD ...data describing sysmods file
   //SMPCNTL  DD *
   SET      BDY(GLOBAL)        /* Set to global zone.   */.
   RECEIVE  SYSMODS            /* Receive only SYSMODs. */
            SOURCEID(MYESO)    /* Specify user-defined
                                  source ID value.      */.
   LIST     SYSMODS            /* Now list the SYSMODs  */
            MCS                /* including SMP MCS     */
            SOURCEID(MYESO)    /* for those SYSMODs just
                                  received.             */.
   /*
   ```

   This makes sure that all available PTFs are ready to be installed. If you find a problem in your system and determine that a PTF must be installed in corrective mode, you have a better chance of having that PTF and all its requisites readily available on the SMPPTS.

   Note: The SOURCEID operand is optional. All PTFs in an ESO are assigned SOURCEID values by ++ASSIGN statements.

Although the procedures for receiving the SYSMOD and the HOLDDATA file are shown as separate jobs, they can be done in one RECEIVE command by specifying both the SYSMODS and HOLDDATA operands. In fact, this is the preferred method and is the default if neither operand is specified.

By following these procedures, you are essentially making a trade-off: system resources as increased DASD space for the SMPPTS against the time the system programmer would spend on searching for the service level with the required PTF and on fixing problems caused by installing PE PTFs.

One important part of this procedure is that the HOLDDATA on each ESO and CBPDO **must be received in chronological order**. SMP/E processes the ++HOLD and ++RELEASE statements in the order in which they are encountered. Therefore, there can be an exposure if you receive the data out of sequence. For instance, the tapes may be set up so that one contains a ++HOLD for a PTF and a subsequent one contains a ++RELEASE for the same PTF. If the tapes are processed in the wrong order, the RELEASE statement is processed first, and then the HOLD statement. As a result, the PTF remains held.

Processing HOLDDATA from PSP Files

Another source of exception SYSMOD data is the PSP file, available through IBMLINK or from the IBM Support Center. For each service level that is applicable to a specific environment, there is a PSP file containing additional HOLDDATA. This file contains all the ++HOLD and ++RELEASE statements applicable to PTFs on either that service level or earlier ones. You should process this PSP file before you install an ESO or before you install a CBPDO tape that includes PTFs from that service level.

When you are ready to install a CBPDO or ESO, you must do the following:

1. Make sure you have received the HOLDDATA from, at minimum, all the CBPDOs and ESOs up to the service level of the tape you are installing. You should receive the HOLDDATA from **all the available tapes** to reduce the amount of data that you have to get from PSP.
2. Use IBMLINK or contact the IBM Support Center to get the latest CORPE PSP file, as well as the PSP file associated with the **latest service level for which you have received HOLDDATA** (not the PSP file for the service level that you are installing).
3. Create a data set containing the ++HOLD statements obtained from IBMLINK or the IBM Support Center.
4. Receive that data set using the SMP/E dialogs or the following sample job.

```
//RECHOLD  JOB 'accounting info',MSGLEVEL=(1,1)
//RECEIVE  EXEC SMPPROC
//SMPHOLD  DD ...data describing your data set
//SMPCNTL  DD *
SET      BDY(GLOBAL)        /* Set to global zone.   */.
RECEIVE  HOLDDATA           /* Receive only exception
                               SYSMOD data.          */.
/*
```

You should also use IBMLINK or contact the IBM Support Center to get PSP information whenever you are installing corrective service. Before installing a PTF in corrective mode, determine the service level on which it was initially shipped. Then contact the IBM Support Center to get the PSP data associated with that service level to determine whether there are any ++HOLD statements for that PTF. If so, process them and install the PTF.

Example of Processing HOLDDATA

Assume you are ready to actually install a CBPDO or ESO. The following example may help you understand the reasons behind the recommendations made in this chapter. In this example:

Exception SYSMODs

1. Table 12 on page 122 shows information on exception SYSMODs. The PTFs and PSP files are as follows:
   a. Column 1 lists the three service levels involved in this example.
   b. Column 2 lists the five SYSMODs in each service level.
   c. Column 3 lists the ++HOLD statements contained on the CBPDO or ESO.

      For simplicity, there are no PE PTFs before the first service level in the example (PUT0201). The exact syntax and APAR numbers for the ++HOLD are not significant for this example.
   d. Column 4 lists the ++HOLD statements contained in the PSP file associated with each of the service levels. The exact syntax and APAR numbers for the ++HOLD are not significant for this example.
2. The SYSMODs have been marked PE as follows:
   - As of the 0201 service level, there were no PTFs in error.
   - Between the 0201 and the 0202 service levels, PTFs UR00002 and UR00003 were marked as PE.
   - Between the 0202 and the 0203 service levels, PTF UR00005 was marked PE.
   - At the 0203 service level, PTF UR00001 was marked as PE.
3. Table 12 shows the contents of each of the files at some time after the creation of the 0203 service level.

Table 12. CBPDO/Service Level/PSP HOLDDATA Example

Service Level	PTFs Per Service Level	PTFs with HOLDDATA	HOLDDATA in PSP File for the Source ID Service Level
PUT0201	UR00001 UR00002 UR00003 UR00004 UR00005		UR00001 UR00002 UR00003 UR00005
PUT0202	UR00006 UR00007 UR00008 UR00009 UR00010	UR00002 UR00003	UR00001 UR00005
PUT0203	UR00011 UR00012 UR00013 UR00014 UR00015	UR00005	UR00001

4. You are now trying to install PTFs at service level 0201. The amount of processing you have to do before installing 0201 service level PTFs depends on what you did with PTFs in 0202 and 0203 service levels.
   - If you have received the HOLDDATA for service levels 0201, 0202, and 0203, you have to process only the one ++HOLD statement for PTF UR00001 from the PSP file for service level 0203.
   - If you have received only the HOLDDATA for service level 0202, you have to process the two ++HOLD statements for PTFs UR00001 and UR00005 from the PSP file for service level 0202.
   - If you decided not to process any data for particular service levels until you are actually ready to install them (that is, if you did nothing with service

level 0202 or 0203), you have to process the four ++HOLD statements for PTFs UR00001, UR00002, UR00003, and UR00005 from the PSP file for service level 0201.

Note: In each case, you used the PSP file associated with the last service level for which you received the HOLDDATA, but if you had kept current in processing the exception SYSMOD files from the service levels, you would have had less information to obtain from the IBM Support Center.

In this example, the number of PTFs and HOLDDATA was small and, thus, the data seems manageable. However, with a real service level with hundreds of PTFs, the amount of manual work involved in getting the ++HOLD statements from IBMLINK or the IBM Support Center, and then keying them into a data set and receiving them could be very time-consuming. So, the cost of the increased DASD space necessary to store the HOLDDATA each month is commonly paid back in increased programmer productivity when the service level is to be installed.

Chapter 9. Creating Cross-Product, Cross-Zone Load Modules: The LINK MODULE Command

This chapter discusses the LINK MODULE command, with an emphasis on when and how to use it.

When to Use LINK MODULE

Products sometimes contain modules from other products. For example, a product may need to:

- Include another product's modules in its load modules. In this case, as long as the two products are in the same zone, SMP/E can automatically include the required modules in the load modules that need them (if the modules reside in the target library as single-CSECT load modules). SMP/E also tracks the inclusion of these cross-product modules in the load modules.
- Update another product's load module with one of its modules. In this case, as long as the two products are in the same zone, SMP/E can automatically relink the load module and include the supplied module. SMP/E also tracks the inclusion of the modules in the cross-product load module.

However, when such products reside in different zones, SMP/E cannot automatically perform the cross-zone link-edits. The LINK MODULE command can be used to perform these cross-zone link-edits as postinstallation steps within SMP/E control. The LINK MODULE command causes the required load modules in one zone to be linked with modules residing in another zone, and tracks this inclusion so that subsequent APPLY and RESTORE processing can automatically maintain the affected load modules.

Notes:

1. When SMP/E processes the LINK MODULE command, it assumes that adding the desired modules to the load modules does not require any changes to the load module definition (that is, the linkage editor control statements or linkage editor attributes). If any such changes are needed, make them through JCLIN before using the LINK MODULE command.
2. For the LINK MODULE command, the SET BOUNDARY command must specify the target zone that contains the LMOD entries for the load modules to be link-edited.
3. There are times when the LINK MODULE command is **not** appropriate to use—generally, for products that are written in a high-level language and, as a result, include modules from libraries (such as compiler libraries) owned by a different product. Your options for installing such a product depend on how the product was packaged.
   - SYSLIB DD statements are used in link-edit steps in order to implicitly include the necessary modules.

     In this case, when you install the product, the implicitly-included modules are automatically linked into the load modules. If the libraries containing those modules are updated, you can use the LINK LMODS command to rebuild the affected load modules. For more information, see the LINK LMODS chapter in *SMP/E Commands*.

- No SYSLIB DD statements are used in link-edit steps in order to implicitly include the necessary modules. In this case, you must use postinstallation link-edit steps outside of SMP/E.

How to Use LINK MODULE

Assume you have installed GDDM and CICS, and some of the GDDM modules must be linked into CICS load modules. GDDM resides in zone GDDTZN, and the zone controlling CICS is CICTZN. Because GDDM and CICS are controlled by different zones, SMP/E does not automatically link the GDDM modules into the CICS load modules when GDDM is installed. The LINK MODULE command can be used instead.

In this example, GDDM module ADMABCD needs to be linked into CICS load module DFHWXYZ. Module ADMABCD is installed in a target library as a single-CSECT load module when GDDM is installed. Therefore, SMP/E can use the target library version of ADMABCD to update CICS load module DFHWXYZ. (If a module does not reside in a target library as a single-CSECT load module, SMP/E uses the related distribution zone copy of the module to update the load module.)

The following commands can be used to have SMP/E install and track the installation of GDDM module ADMABCD in the CICS load module:

```
SET   BDY(CICTZN)         /* Target zone for CICS.      */.
LINK  MODULE(ADMABCD)     /* Link module ADMABCD        */
      FROMZONE(GDDTZN)    /* residing in zone GDDTZN    */
      INTOLMOD(DFHWXYZ)   /* into load module DFHWXYZ.  */.
```

These commands cause GDDM module ADMABCD to be linked into CICS load module DFHWXYZ. SMP/E also adds cross-zone subentries to the affected entries:

- An XZLMOD subentry is added to the ADMABCD MOD entry in target zone GDDTZN so that if ADMABCD is updated, it can be automatically replaced in the CICS load module.

  Note: The CICS load module is automatically updated **only** if the XZLINK subentry was previously set to AUTOMATIC in the TZONE entry for zone CICTZN. Here is an example of the commands that can be used to do this:

  ```
  SET  BDY(CICTZN)        /* Target zone for CICS.    */.
  UCLIN.
  ADD  TZONE(CICTZN)      /* Update TZONE entry       */
       XZLINK(AUTOMATIC). /* to do automatic links.   */.
  ENDUCL.
  ```

- An XZMOD subentry is added to the CICS DFHWXYZ LMOD entry in target zone CICTZN to indicate that:
  - DFHWXYZ now contains ADMABCD.
  - Any updates for ADMABCD should be accepted **only** from zone GDDTZN.
- TIEDTO subentries are added to the TZONE entries for CICTZN and GDDTZN to indicate that there is a relationship between modules and load modules in these zones.

For more information on the LINK MODULE command and cross-zone updating during APPLY and RESTORE processing, see the LINK MODULE chapter in *SMP/E Commands*.

Chapter 10. Displaying the Data Managed by SMP/E: The LIST Command

This chapter discusses the LIST command. After an introduction, it discusses these topics:
- Listing all the SMP/E data
- Listing by specific entry type
- Listing specific entries
- Listing by FMID or FMIDSET
- Listing to compare two zones

Introduction

The SMP/E database contains much information that is useful to you at certain times. For instance, when a problem is encountered in your system:
- You need to know the functional and service level of the module with the error.
- You may also want to know when that module was last changed: a recent change may have caused the problem.
- After reporting the problem, you can start working with the IBM Support Center to debug the problem. They may want to know the status of other specific PTFs: have you installed them; are they available on your system; is anything stopping them from being installed?
- After identifying the problem in the module, you may want to know whether any other parts of the system are affected by that module.

All of this information is available in the SMP/E database. You can obtain the information by using the SMP/E dialogs as you debug the problem. You can also use the SMP/E LIST command to create hardcopy listings of the information.

With the LIST command, you can display all entries of a specified type or specific entries. The following sections demonstrate the flexibility of the LIST command. For a complete description of all the LIST command operands, see The LIST Command in *SMP/E Commands*.

Listing All the SMP/E Data

If you encounter a problem with your system, the SMP/E data describing that system can be very important in diagnosing the problem. This information can be obtained by using the SMP/E dialogs during the debugging process. However, if the system is not running, the information is not available unless you have periodically listed the SMP/E data for the system.

Therefore, it is advisable to list all the data for each of your systems and save the hardcopy listing. The data can be listed by individual zone. The following is an example of a job for listing all the entries in zone TGT01.

```
//LIST     JOB 'accounting info',MSGLEVEL=(1,1)
//LIST     EXEC SMPPROC
//SMPCNTL  DD  *
SET      BDY(TGT01)         /* Set to desired zone.     */.
LIST                        /* List all entries         */
```

```
        XREF              /* with XREF option to show
                             additional relationships
                             between entries.        */.
/*
```

Because the global zone contains data used to process each of the target zones and distribution zones, you may want to list that data more often. The following job lists all the data in the global zone, including the SYSMOD entries and the MCS entries:

```
//LIST     JOB 'accounting info',MSGLEVEL=(1,1)
//LIST     EXEC SMPPROC
//SMPCNTL  DD  *
SET      BDY(GLOBAL)        /* Set to global zone.      */.
LIST                        /* List all entries.        */.
/*
```

If you do not require a listing of the SYSMOD and MCS entries, you can use the LIST operands that enable you to list only specific entry types. For additional information, see “Listing by Specific Entry Type”.

SMP/E also provides support to list all the entries for all the zones defined in the GLOBALZONE entry. This enables you to display all data for all systems with one SMP/E command. Here is an example of this option:

```
//LIST     JOB 'accounting info',MSGLEVEL=(1,1)
//LIST     EXEC SMPPROC
//SMPCNTL  DD  *
SET      BDY(GLOBAL)        /* Set to global zone.      */.
LIST                        /* List all entries         */
         ALLZONES           /* for all zones.           */.
/*
```

Notes:

1. The ALLZONES operand should be used with caution; it may produce a large amount of output.
2. This function can also be qualified by other LIST operands to limit the entries listed from each zone. For example, the following job will list only the SYSMOD entries from all zones:

```
//LIST     JOB 'accounting info',MSGLEVEL=(1,1)
//LIST     EXEC SMPPROC
//SMPCNTL  DD  *
SET      BDY(GLOBAL)        /* Set to global zone.      */.
LIST     SYSMOD             /* List the SYSMOD entries  */
         ALLZONES           /* for all zones.           */.
/*
```

Listing by Specific Entry Type

At times, you may need to have a listing of all entries of a certain type. For example:

- You may want to display all the DDDEF entries for a particular target zone or distribution zone.
- You may want to list all the OPTIONS and UTILITY entries in the global zone so you do not duplicate an existing entry.

SMP/E supports various operands on the LIST command that you can use to list all the entries for one or more entry types. The following job shows how to use the DDDEF, OPTIONS, and UTILITY operands on the LIST command to do this:

```
//LIST     JOB 'accounting info',MSGLEVEL=(1,1)
//LIST     EXEC SMPPROC
//SMPCNTL  DD  *
SET      BDY(TGT1)          /* Set to target zone       */.
LIST     DDDEF              /* List all DDDEF entries.  */
                            /*                          */.
SET      BDY(DLIB1)         /* Set to DLIB zone         */.
LIST     DDDEF              /* List all DDDEF entries.  */
                            /*                          */.
SET      BDY(GLOBAL)        /* Set to global zone.      */.
LIST     OPTIONS            /* List all options         */
         UTILITY            /* and utility entries.     */
                            /*                          */.
/*
```

For a complete list of all the operands corresponding to the various entry types, see The LIST Command in *SMP/E Commands*.

Note: Not all entry types are valid for each zone type. For example, requesting a listing of the OPTIONS entries in a target zone results in an error, because OPTIONS entries exist only in the global zone. For a summary of the types of entries contained in each type of zone, see Table 2 on page 55 and Table 3 on page 56.

Sometimes, the various entry-type operands can be further qualified to process only a subset of all existing entries. The most common type for which this can be done is the SYSMOD entries. There are numerous operands to qualify SYSMOD entries. The LIST Command in *SMP/E Commands* describes all of them in detail.

One of the most common uses of this function is to determine the functions (or FMIDs) that have been installed. The following job can be used to list:
- The function SYSMODs installed on TGT1
- Those PTFs that have been applied to TGT1 but not yet accepted to DLIB1

```
//LIST     JOB 'accounting info',MSGLEVEL=(1,1)
//LIST     EXEC SMPPROC
//SMPCNTL  DD  *
SET      BDY(TGT1)          /* Set to target zone.      */.
LIST     SYSMOD             /* List the SYSMOD entries  */
         FUNCTIONS          /* for function SYSMODs.    */
                            /*                          */.
LIST     SYSMOD             /* List the SYSMOD entries  */
         PTFS               /* for PTF type SYSMODs     */
         NOACCEPT(DLIB1)    /* not yet accepted.        */
                            /*                          */.
/*
```

Listing Specific Entries

When you encounter a problem in your system and contact the IBM Support Center to resolve the problem, you may be asked to provide very specific information. For example:
- What is the service level of the module in which the problem was reported?
- Are there any USERMODs for the module?
- Do you have a specific PTF installed?

If you have a complete, current listing of all the entries in your system, you can get this information from that listing. You can also get it through the SMP/E dialogs while you are talking to the IBM Support Center.

LIST Command

SMP/E also provides additional LIST functions that you can use to display only specified entries. This is done by allowing you to specify a list of entry names (in parentheses) after each of the entry-type operands. For example, assume that you need to know the function and service level for modules GIMMPDRV and GIMMPIO and if SYSMOD UR12345 has been installed. The following job can be used:

```
//LIST     JOB 'accounting info',MSGLEVEL=(1,1)
//LIST     EXEC SMPPROC
//SMPCNTL  DD  *
SET      BDY(TGT1)          /* Set to target zone.      */.
LIST     MOD(GIMMPDRV       /* List these two modules   */
             GIMMPIO)       /*                          */
         SYSMOD(UR12345)    /* and this SYSMOD.         */
                            /*                          */.
/*
```

You receive a listing of the required information for the two modules. If SYSMOD UR12345 was installed, it should be listed; otherwise, you receive a message saying that the entry was not found (meaning it has not been installed).

Another common use for this function is to list the cover letters for specific PTFs. The following job shows an example of a job for listing the cover letters for PTFs UR00001, UR00002, and UR00003:

```
//LIST     JOB 'accounting info',MSGLEVEL=(1,1)
//LIST     EXEC SMPPROC
//SMPCNTL  DD  *
SET      BDY(GLOBAL)        /* Set to global zone.      */.
LIST     MCS(               /* List cover letters       */
             UR00001        /* for these three PTFs.    */
             UR00002
             UR00003
            )
         .
/*
```

Listing by FMID or FMIDSET

Frequently, you deal with one area of the system at a time and would like to see all the information relating to that one area. You can use the FORFMID operand in conjunction with the various entry-type operands to limit the entries processed. Here is an example listing:

- All the entries associated with function HXY1100
- All the MAC entries associated with function HXZ2100
- All the SYSMOD and module entries associated with either function JXY1123 or JXY1124

```
//LIST     JOB 'accounting info',MSGLEVEL=(1,1)
//LIST     EXEC SMPPROC
//SMPCNTL  DD  *
SET      BDY(TGT1)          /* Set to target zone.      */.
LIST                        /* List all entries         */
         FORFMID(HXY1100)   /* for this FMID.           */
                            /*                          */.
LIST     MAC                /* List only the macros     */
         FORFMID(HXZ2100)   /* for this FMID.           */
                            /*                          */.
LIST     SYSMOD             /* List SYSMOD entries      */
         MOD                /* and MOD entries          */
```

```
        FORFMID(JXY1123    /* for this FMID           */
                JXY1124)   /* or this FMID.           */
                           /*                         */.
/*
```

Note: The names within the FORFMID operand can be names of FMIDSET entries. In this case, SMP/E lists all the entries associated with any of the FMIDs defined in the FMIDSET entry.

Listing to Compare Two Zones

If you have multiple zones, you may sometimes want to determine the functional and service differences between them. The LIST command provides you with this capability.

Note: You can also use the REPORT SYSMODS command to compare the SYSMOD content of two zones. Besides telling you which SYSMODs are installed in one zone but not in a second, REPORT SYSMODS also indicates which of the uninstalled SYSMODs are applicable to the second zone and generates commands you can run to install the SYSMODs in the second zone. For more information, see Chapter 15, “Comparing the SYSMODs Installed in Two Zones: The REPORT SYSMODS Command”, on page 145.

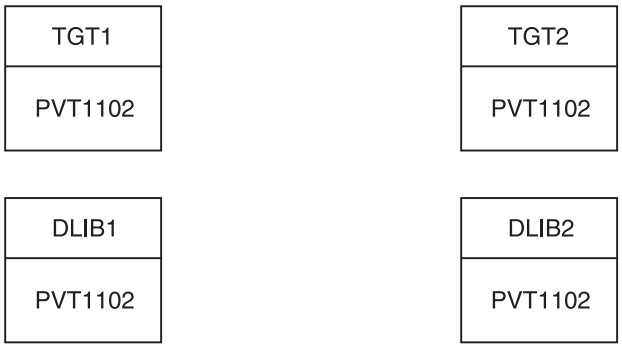
One possibility might be that you have two products at different service levels. The product at the lower service level works, and the product at the higher service level does not work. You might use LIST to compare the zones for the two systems and to determine what is causing the problem.

This example compares two target zones, TGT1 and TGT2. The commands in the following example perform the comparison for you:

```
//LIST     JOB 'accounting info',MSGLEVEL=(1,1)
//LIST     EXEC SMPPROC
//SMPCNTL  DD  *
SET      BDY(TGT1)          /* Set to target zone TGT1. */.
LIST     SYSMODS            /* List the SYSMODs         */
                            /* in zone TGT1             */
         NOAPPLY(TGT2)      /* that have not been       */
                            /* applied to TGT2.         */
                            /*                          */.
SET      BDY(TGT2)          /* Set to target zone TGT2. */.
LIST     SYSMODS            /* List the SYSMODs         */
                            /* in zone TGT2             */
         NOAPPLY(TGT1)      /* that have not been       */
                            /* applied to TGT1.         */
                            /*                          */.
/*
```

By comparing the two resulting listings, you can see the differences between the two zones.

In this second example, the same product is installed in different zones. You want to compare the service to make sure both copies of the product are at the same level. For example, assume that product PVT1102 is installed in two target zones and two distribution zones:



You want to make sure that PVT1102 is at the same service level in all the zones. To do this, you can use the LIST command and compare which SYSMODs are installed in which zones.

To compare the service levels of product PVT1102 in the two distribution zones, you can use the commands in the following example:

```
//LIST     JOB 'accounting info',MSGLEVEL=(1,1)
//LIST     EXEC SMPPROC
//SMPCNTL  DD  *
SET    BDY(DLIB1)          /* Set to DLIB1.        */.
LIST   SYSMOD              /* List SYSMODs         */
       FORFMID (PVT1102)   /* for PVT1102          */
       NOACCEPT (DLIB2)    /* in DLIB1, not DLIB2. */.
SET    BDY(DLIB2)          /* Set to DLIB2.        */.
LIST   SYSMOD              /* List SYSMODs         */
       FORFMID (PVT1102)   /* for PVT1102          */
       NOACCEPT (DLIB1)    /* in DLIB2, not DLIB1. */.
```

Similarly, to compare the service records for the target zone copies of PVT1102, you can use LIST with the NOAPPLY operand.

Summary

The LIST command enables you to:

- Compare two target zones using the NOAPPLY operand.
- Compare two distribution zones using the NOACCEPT operand in place of the NOAPPLY operand.
- Compare a target zone and a distribution zone using both the NOAPPLY and NOACCEPT operands.

This gives you the ability to compare all combinations of zone types, keeping in mind that the zone for the NOAPPLY operand must be a target zone, and that the zone for the NOACCEPT operand must be a distribution zone.

Chapter 11. Changing the Data SMP/E Manages: The UCLIN Command

This chapter discusses the UCLIN command, with an emphasis on how and when to use it.

Introduction

The SMPCSI and associated data sets contain basically two types of information:

- Information added as a result of installing SYSMODs. Generally, this type is managed completely by SMP/E; that is, appropriate entries are added, changed, or deleted as SYSMODs are installed. You need not make any modification to the database to record this information. You may, however, need to make changes to do the following:
  - Record changes made outside SMP/E
  - Delete information no longer required
  - Recover from an SMP/E or system error
- Information added by you to control the installation of SYSMODs. You must manually add this information to the database before processing any SYSMODs. You may later need to modify the information to reflect new processing information.

The UCLIN command helps you to make these changes.

To use UCLIN effectively, you should have a detailed understanding of how it works and what it can do. The UCLIN Command in *SMP/E Commands* and SMP/E data-set entries in the *SMP/E Reference* manual provide that level of detail. Read those chapters before trying to run any UCLIN commands. The following sections describe, at a very high level, what UCLIN is.

When to Use UCLIN

UCLIN is a very powerful function that must be used with **extreme caution**. You can use UCLIN to modify almost all the data in the SMP/E database. When you are modifying an entry, SMP/E makes sure the data within that one entry is consistent—that is, that the result could have occurred during normal SMP/E processing. However, no checking is done to make sure the resulting entry is consistent with other related entries in the database.

For example, you can use UCLIN to delete a UTILITY entry in the global zone without SMP/E detecting any error condition. However, if there is an OPTIONS entry within the global zone that refers to the deleted UTILITY entry, an error occurs when you attempt to use that OPTIONS entry. This is a very simple example of inconsistent data across entries that does not result in a serious error. UCLIN modifications to other entries, such as element, LMOD, or SYSMOD, may not be detected as error conditions during processing, but may cause incorrect processing, such as failing to link a module, updating the wrong library, or installing a SYSMOD that should not be installed.

In general, consider the following before making the UCLIN update:

1. Determine whether there is a better method of obtaining the same result. Table 13 on page 134 shows where to find more information about alternatives

to UCLIN.

Table 13. Alternatives to UCLIN

To Do This without UCLIN:	Look Here for More Information:
Change a common subentry in several DDDEF or UTILITY entries in the same zone	The ZONEEDIT Command in *SMP/E Commands*
Update the cross-zone subentries of the MOD, LMOD, and TARGETZONE entry	The ZONEEDIT Command in *SMP/E Commands*
Add, rename, or delete a zone	*SMP/E Commands*: Chapters on the following commands: • **Adding a zone:** To add a zone, you can use the following commands, depending on the particular situation: ZONECOPY and ZONERENAME ZONEEXPORT, UCLIN for the ZONEINDEX, and ZONEIMPORT • **Renaming a zone:** To rename a zone, you must use the following command: ZONERENAME • **Deleting a zone:** To delete a zone, you can use the following commands, depending on the particular situation: ZONEDELETE or ZONEEXPORT
Change the structure of your system (For example, to add, delete, move, or rename elements or their aliases)	*z/OS Packaging Rules*: Section on how to avoid UCLIN by using the appropriate MCSs and operands

2. If you are not the originator of the UCLIN, make sure you understand **exactly** what is being done and why. If you are not sure, find out before making the update.
3. Make sure the UCLIN is being done in the correct sequence in the process—before or after the installation of the SYSMOD.
4. Make sure all the data is correct.
5. List the entry before changing it. This makes sure you know what the original entry looked like in case an error is reported during the UCLIN or the modification causes an error.
6. After you have done all of these steps, if you have been given directions for installing the UCLIN (for example, in the PTF cover letter or in the program directory for a new function), follow those directions.

How to Use UCLIN

UCLIN is used to update the entries in the SMP/E database, just as the SPZAP utility is used to update the system libraries. That is, it enables you to:
- Add new information
- Delete existing information
- Replace existing information with new information

Some UCLIN functions will not work for certain entries or data sets. The chapters on the UCLIN command in *SMP/E Commands* and SMP/E data-set entries in

SMP/E Reference provide detailed information on which entries may be modified for each data set, what data within each entry may be modified, and the exact syntax for each entry and data item.

The general format for UCLIN statements is:

```
SET     BDY(xxxxxxx)        /* Set to correct zone.     */.
UCLIN                       /* Marks start of UCLIN.    */.
  ...                       /*   UCL                    */
  UCL statements            /*   statements             */
  ...                       /*   to make modifications. */
ENDUCL                      /* Marks end of UCLIN.      */.
```

The general format of each UCL statement is as follows:

```
ADD|REP|DEL                 /* Type modification        */
   type(name)               /* Entry type and name      */
   operand                  /*                          */
   operand                  /*     Optional             */
   operand                  /*     operands             */
   operand                  /*                          */
                            /* End of one UCL statement */.
```

Where:

ADD|REP|DEL
specifies the action to be taken on the entry or operands specified.

In general, ADD means add the entry or operand only if it is not already present; REP means replace the operand if it is already present, otherwise add it; and DEL means delete the entry or operand if it exists. For a more detailed description of the functions provided by ADD, REP, and DEL, see the chapter on the UCLIN command in the *SMP/E Commands* manual.

type(name)
specifies the entry type (such as MOD, MAC, SRC, data element type, SYSMOD) and name of the entry (such as GIMMPDRV, HELP, MYSRCMOD, MYCLIST, UR12345).

operand
specifies the individual data items in the entry that are to be modified.

The data items can be:

- Single operands, such as RENT or REUS or COPY
- Single value subentries, such as DISTLIB(AOS12), where only one value can be placed within the parentheses
- Multiple value subentries, such as LMOD(LMOD01,LMOD02,LMOD3) or MAC(MAC01,MAC02), where more than one value can be specified within the parentheses

Chapter 12. Identifying Cross-Zone Requisites: The REPORT CROSSZONE Command

This chapter contains information about using the REPORT CROSSZONE command to check for requisites between zones. It discusses these topics:

- How to define a ZONESET
- How to run the REPORT CROSSZONE command
- How to install the SYSMODs, using the output from the REPORT CROSSZONE command

Introduction

Your system may contain products that are packaged and installed separately, but which have service level or interface dependencies. For example, an interface error in one product may require a change to another product that used the interface. When this happens, a unique PTF is generated for each product. The relationship between the PTFs can be specified in a conditional requisite (++IF) modification control statement in the PTFs. If you have completed the steps listed in "Specifying Automatic Cross-Zone Requisite Checking" on page 73, SMP/E automatically checks the requisites during APPLY, ACCEPT, and RESTORE processing, whether the products are in the same zone or in separate zones. However, if you have **not** completed these steps and the products are in separate zones, the requisites are not automatically checked in any other zones. In this case, to make sure these requisites are installed where they are needed, you must:

1. Define a set of zones to be checked for conditional requisites. This is done with the ZONESET entry in the global zone.

   Note: All the zones in the ZONESET must be defined by ZONEINDEX subentries in the same global zone as the ZONESET entry.
2. Run the REPORT CROSSZONE command to get a list of the SYSMODs that must be installed and the zones where they are needed.
3. Install the SYSMODs in the indicated zones.

Note: If you just want to check service levels for products, you should use the REPORT SYSMODS command or the LIST command. See Chapter 15, "Comparing the SYSMODs Installed in Two Zones: The REPORT SYSMODS Command", on page 145 and "Listing to Compare Two Zones" on page 131 for more information.

Defining a ZONESET

To tell SMP/E which zones it should check for missing requisites, you must define a global zone ZONESET entry. You may have one or more ZONESETs to describe groups of products that might have dependencies on each other.

For example, assume you have a system that supports two products, ABC and AYZ, that have dependencies on one another. You might have one zone, BASEABC, for the base ABC function, and another zone, PRODABC, for a dependent function. Likewise, you might have a zone BASEXYZ for the base XYZ function, and another zone, PRODXYZ, for a dependent function. The dependent functions are

different versions of the same product, and they must be synchronized with each other and with their base functions. You can set up two ZONESETs to help keep these products at the same service level.

These are the commands you can use to define the ZONESETs:

```
//UCL      JOB 'accounting info',MSGLEVEL=(1,1)
//UCL      EXEC SMPPROC
//SMPCNTL  DD  *
SET      BDY(GLOBAL)      /* Set to global zone.   */.
UCLIN                     /*                       */.
ADD      ZONESET(ZONEA)   /* Create ZONESET ZONEA. */
         ZONE(BASEABC,    /* Include these zones.  */
              PRODABC,    /*                       */
              PRODXYZ)    /*                       */
         .
ADD      ZONESET(ZONEX)   /* Create ZONESET ZONEX. */
         ZONE(BASEXYZ,    /* Include these zones.  */
              PRODXYZ,    /*                       */
              PRODABC)    /*                       */
         .
ENDUCL                    /*                       */.
/*
```

When you define a ZONESET, remember:

- Each zone in a ZONESET must also be defined in the global zone.
- Each zone in a ZONESET must be defined in the same global zone. They cannot be defined in global zones that are in different CSI data sets.
- A zone can be part of more than one ZONESET.
- A ZONESET can contain both target and distribution zones.

For more information on defining the ZONESET entry, see *SMP/E Reference*.

Running the REPORT CROSSZONE Command

After you have defined the appropriate ZONESET, you can run the REPORT CROSSZONE command to get a list of all the SYSMODs that must be installed and which zones in the ZONESET need them. This list is the Cross-Zone Requisite SYSMOD report. It identifies which of the needed SYSMODs must be received and which SYSMODs caused the needed SYSMODs to be listed. Besides the report, SMP/E writes commands to the SMPPUNCH data set. You can use them to install the SYSMODs in the appropriate zones.

If all the zones in a ZONESET are of the same type, SMP/E determines the type of report to be generated. For example, a ZONESET containing only target zones results in a Cross-Zone Requisite SYSMOD report for APPLY processing. On the other hand, your ZONESET can be mixed; that is, it may contain both target and distribution zones. If this is the case, you need to specify which type of Cross-Zone report you want generated.

You can limit which SYSMODs SMP/E reports on by specifying any combination of these operands on the REPORT CROSSZONE command:

- **FORZONE:** to list only SYSMODs needed in specific zones in the ZONESET
- **FORFMID:** to list only SYSMODs needed for specific FMIDs in the ZONESET zones
- **DLIBZONE** or **TARGETZONE:** to tell SMP/E which zones you want a report on (if the zones contained in the zone set specified by ZONESET are mixed)

Note: DLIBZONE and TARGETZONE are mutually exclusive operands.

For more information on the REPORT CROSSZONE command, see the *SMP/E Reference* manual.

To continue the example for this chapter, assume you applied a number of SYSMODs to the zones in ZONESET ZONEA. Some of these SYSMODs named conditional requisites. To find out which zones are affected by these requisites, you can use the following commands:

```
//REPORT   JOB 'accounting info',MSGLEVEL=(1,1)
//REPORT   EXEC SMPPROC
//SMPCNTL  DD  *
SET    BDY(GLOBAL)      /* Set to global zone.  */.
REPORT CROSSZONE        /* Report on            */
       ZONESET(ZONEA)   /* ZONESET ZONEA.       */.
/*
```

Installing the SYSMODs

Once you have the Cross-Zone Requisite SYSMOD report, you can install the missing SYSMODs in the zones where they are needed. Follow these steps:

1. Receive any SYSMODs that have not yet been received.
2. Install the SYSMODs in the zones where they are needed. If you have the SMPPUNCH output from the REPORT CROSSZONE command, you can use that.
3. Rerun the REPORT command using the same ZONESET to check the results of installing the new SYSMODs.
4. Receive and install any additional SYSMODs that are needed.

For the example in this chapter, follow these steps for ZONESET ZONEA and ZONESET ZONEX, because both contain zone PRODXYZ. You can continue to run the REPORT CROSSZONE command and install SYSMODs until all the needed SYSMODs have been installed in both ZONESETs.

Chapter 13. Identifying Installed SYSMODs Affected by Error Holds: The REPORT ERRSYSMODS Command

This chapter contains information about using the REPORT ERRSYSMODS command to check for installed SYSMODs affected by error holds that were subsequently received. It discusses these topics:

- How to run the REPORT ERRSYSMODS command
- How to use output from the REPORT ERRSYSMODS command for installing the SYSMODs

Introduction

In the course of maintaining your system, you can apply or accept service and later receive HOLDDATA that affects that service. If any of that HOLDDATA was for an error reason ID, you should install a resolving SYSMOD so that the error does not occur on your system. However, SMP/E does not automatically write any reports during RECEIVE processing to help you do this. To see if any installed SYSMODs are affected by error holds that were subsequently received, you must:

1. Run the REPORT ERRSYSMODS command to get a list of the SYSMODs that must be installed and the zones where they are needed.
2. Install the resolving SYSMODs in the indicated zones.
3. Determine whether any resolving SYSMODs are available for held SYSMODs.

Running the REPORT ERRSYSMODS Command

After you have received HOLDDATA, you can run the REPORT ERRSYSMODS command to get a list of all the SYSMODs that are affected by any unresolved error holds. This list is the Exception SYSMOD report, which also tells whether any resolving SYSMODs have been received for these holds and whether any of the resolving SYSMODs have error holds against them. Besides the report, SMP/E writes commands to the SMPPUNCH data set. You can use them to install the resolving SYSMODs in the appropriate zones.

You can limit which HOLDDATA SMP/E reports on by specifying any combination of these operands on the REPORT ERRSYSMODS command:

- **BEGINDATE:** to check HOLDDATA entries for error reason IDs that were received by SMP/E on or after the specified date
- **ENDDATE:** to check HOLDDATA entries for error reason IDs that were received by SMP/E on or before the specified date
- **FORFMID:** to list only SYSMODs owned by specific FMIDs

For more information on the REPORT ERRSYSMODS command, see *SMP/E Commands*.

Here is an example of when you might want to use the REPORT ERRSYSMODS command. Assume that you have just received some HOLDDATA, and you need to know whether it affects any of the SYSMODs you have already accepted into distribution zone DZONE1. You can use the following commands:

REPORT ERRSYSMODS Command

```
//REPORT   JOB 'accounting info',MSGLEVEL=(1,1)
//REPORT   EXEC SMPPROC
//SMPCNTL  DD  *
SET    BDY(GLOBAL)         /* process global zone     */.
REPORT ERRSYSMODS          /* report on exception     */
       ZONES(DZONE1)       /* SYSMODs in this zone    */
       BEGINDATE(01 01 02) /* for HOLDDATA received   */
       ENDDATE(02 01 02)   /* between these dates     */.
```

Installing the SYSMODs

Once you have the Exception SYSMOD report, you can install the resolving SYSMODs in the zones where they are needed. Follow these steps:

1. If any resolving SYSMODs are held, run REPORT ERRSYSMODS, specifying the global zone, to see if any SYSMODs have been received that resolve the additional holds for the resolving SYSMODs.
2. If the Exception SYSMOD report for the global zone shows resolving SYSMODs for the additional holds, edit the SMPPUNCH output to add the new resolving SYSMODs and to update the selection list so the held resolving SYSMODs will be processed.
3. Use the SMPPUNCH output to install the resolving SYSMODs in DZONE1.

Chapter 14. Listing the Source IDs in a Zone: The REPORT SOURCEID Command

This chapter contains information about using the REPORT SOURCEID command to list the source IDs assigned to SYSMODs in a given zone or ZONESET. It discusses these topics:

- How to run the REPORT SOURCEID command
- How to list SYSMODs using the output from the REPORT SOURCEID command

Introduction

In the course of maintaining your system, you may need to find out which source IDs are assigned to SYSMODs in a given zone. For example, assume you install service using CBPDOs, which assign source IDs to the service SYSMODs they contain. You can use the REPORT SOURCEID command to determine the latest service level you have installed in a particular zone. To determine the service level based on source IDs, follow these steps:

1. Run the REPORT SOURCEID command to get a list of which source IDs are assigned to SYSMODs in a given zone.
2. If desired, list the SYSMOD entries for the SYSMODs with those source IDs.

Running the REPORT SOURCEID Command

You can use the REPORT SOURCEID command to get a list of all the source IDs assigned to SYSMODs in a given zone. This list is the SOURCEID report, which may also indicate which SYSMODs these source IDs are assigned to. Besides the report, SMP/E writes commands to the SMPPUNCH data set, which you can use to list the SYSMODs. For details on the REPORT SOURCEID command, see *SMP/E Commands*.

Here is an example of when you might want to use the REPORT SOURCEID command. Assume you want to find out which source IDs are associated with SYSMODs in target zone TGT1, and you want to know which SYSMODs each source ID is assigned to. You can use the following commands:

```
SET BDY(GLOBAL).
REPORT SOURCEID
       ZONES(TGT1)
       SYSMODIDS.
```

Listing the SYSMODs

If you want more information about the SYSMODs that are assigned the source IDs shown in the SOURCEID report, you can list the related SYSMOD entries. The SMPPUNCH output produced by the REPORT SOURCEID command contains the LIST SYSMOD SOURCEID(...) commands needed to list the SYSMODs for the source IDs in the SOURCEID report. You can tailor the SMPPUNCH output to list the SYSMODs in which you are interested, and run the commands to list the desired SYSMODs.

Chapter 15. Comparing the SYSMODs Installed in Two Zones: The REPORT SYSMODS Command

This chapter contains information about using the REPORT SYSMODS command to check if SYSMODs installed in one zone are applicable to and installed in a second zone. It discusses these topics:

- How to run the REPORT SYSMODS command
- How to install SYSMODs using the output from the REPORT SYSMODS command

Introduction

In the course of maintaining your system, you may need to compare the function and service level of two zones. For example, if you have installed the same products in different zones, you may want to make sure that both copies of these products are at the same level. SMP/E does not do this kind of checking automatically. To compare the SYSMODs installed in two zones, you must:

1. Run the REPORT SYSMODS command to get a list of the SYSMODs that are installed in the first zone and that are applicable to the second zone but are not yet installed there.
2. Install the applicable SYSMODs in the second zone.

Running the REPORT SYSMODS Command

You can run the REPORT SYSMODS command to get a list of all the SYSMODs that are installed in one zone and not in a second. This list is the SYSMOD Comparison report, which also tells which of these SYSMODs are applicable to the second zone, shows if the SYSMODs have been received, and lists any source IDs associated with the SYSMODs. Besides the report, SMP/E writes commands to the SMPPUNCH data set, which you can use to install the SYSMODs. For details on the REPORT SYSMODS command, see *SMP/E Commands*.

Here is an example of when you might want to use the REPORT SYSMODS command. Assume you have two z/OS systems. The target zones that control these systems are TGZONE1 and TGZONE2, and they are serviced from the same global zone. You want to determine which SYSMODs are installed in TGZONE1 and are not installed in, but are applicable to, TGZONE2. You can use the following commands:

```
SET    BDY(GLOBAL)          /* process global zone     */.
REPORT SYSMODS              /* report on SYSMODs       */
       INZONE(TGZONE1)      /* input zone TGZONE1      */
       COMPAREDTO(TGZONE2)  /* comparison zone TGZONE2 */.
```

Installing the SYSMODs

Once you have the SYSMOD Comparison report, you can install the applicable SYSMODs in the zone where they are needed. Follow these steps:

1. Research the report to determine which of the identified SYSMODs you want to install into the comparison zone.

2. Find and receive any applicable SYSMODs that were not available and that you want to install. The source ID in the report identifies some possible sources for obtaining the SYSMODs.
3. Tailor the SMPPUNCH output to install the set of SYSMODs that you deem appropriate; then run the commands to install the desired SYSMODs.

Chapter 16. Building a User Modification

This chapter discusses steps and considerations for building a user modification (USERMOD). It provides the following information:

- How to choose between building a SYSMOD as a USERMOD and building it as a function
- How to create modification control statements
- Examples of USERMODs

Choosing between a USERMOD and a Function SYSMOD

Software products available from IBM provide you with many functions. However, these functions may not always exactly meet your processing requirements. Many of these products provide interfaces, such as user exit routines or dummy modules, that you can use to customize the functions to your needs. Sometimes, however, you may need to change a function substantially. You can do this by either:

- Constructing a user modification as USERMODs to an existing function
- Constructing an additional function SYSMOD

Although you might be able to make these changes without SMP/E, there are advantages to creating them either as USERMODs or as function SYSMODs so that SMP/E can install them.

When SMP/E installs the changes, it does the following:

- Keeps a record of the changes
- Reports any intersections with SYSMODs provided by IBM
- Ensures that the changes are not regressed
- Ensures that the changes are installed properly in the correct libraries
- Lets you remove the changes if there are problems

Before creating your changes, you must decide whether to build a USERMOD type SYSMOD or a function SYSMOD. Table 14 lists considerations that will help you make this decision.

Table 14. Comparison of USERMODs and Function SYSMODs

USERMOD	Function SYSMOD
Provides changes for elements owned by an existing function. May provide new elements for an existing function.	Provides new elements or new element versions for a new function.
SMP/E reports on changes attempted by PTFs, APARs, or USERMODs.	SMP/E does not report on changes attempted by PTFs, APARs, or USERMODs.
Other SYSMODs for the same function can update the elements.	Because the SYSMOD owns the element, SYSMODs for other functions cannot update the element without also changing the owner of the element.
Better for small changes that affect only a few elements.	Better for major additions to the system.

Creating the MCSs

This section describes some of the considerations for building the MCSs for a USERMOD SYSMOD. See *z/OS Packaging Rules* for guidelines on when to use USERMODs and SMP/E Modification Control Statements in *SMP/E Reference* for more information on MCS syntax.

The ++USERMOD MCS

The ++USERMOD statement identifies this SYSMOD as a USERMOD and assigns a 7-character identifier to the SYSMOD.

The format of the ++USERMOD statement is:

```
++USERMOD(sysmod_id)          /*                        */.
```

The ++VER MCS

The ++VER statement is necessary in all SYSMODs. It describes the environment necessary for installing the SYSMOD.

The general format of the ++VER statement follows:

```
++VER                       /* Environment MCS          */
         (srel)             /* System and release ID    */
     FMID(aaaaaaa)          /* Functional area          */
     PRE (                  /*                          */
          bbbbbbb bbbbbbb   /* Prerequisite PTFs        */
          bbbbbbb bbbbbbb   /*                          */
         )                  /*                          */
     REQ (                  /*                          */
          ccccccc ccccccc   /* Other related USERMODs   */
          ccccccc ccccccc   /*                          */
         )                  /*                          */
     SUP (                  /*                          */
           ddddddd dddddddd /* Other USERMODs incorp-   */
           ddddddd dddddddd /* orated into this one     */
         )                  /*                          */.
```

Specifying the Proper System Release

The SREL value (*srel*) must be one of those defined as an SREL subentry in the TARGETZONE entry. If the USERMOD is a change to an IBM product, the SREL should correspond to the SREL value specified in the IBM product that currently owns the elements within this SYSMOD.

Specifying the FMID Value

If any element is owned by an FMID different from that specified in the ++VER statement, that element is not selected for installation during APPLY or ACCEPT processing, and message GIM45401W is issued. This condition is a SYSMOD construction error. SMP/E supports a SYSMOD construction that assumes that this condition occurs regularly and that the SYSMOD contains an ++IF statement specifying another SYSMOD that supplies the proper functional version of the element.

Note: As was explained earlier, it is a good idea to construct a USERMOD so that each SYSMOD contains only one element. This construction method eliminates this problem.

Specifying the Proper Requisites

When you specify requisite SYSMODs, you are defining two kinds of relationships:

- The relationship of your SYSMOD to previous versions of the element
- The relationship of your SYSMOD to other SYSMODs currently on the system

The following text describes how you can define these relationships.

Relationships to Earlier Versions of the Elements:

1. If the element entry in your target zone has an RMID value different from its FMID value, ensure it is a prerequisite of the USERMOD fix; that is, make sure the *bbbbbbb* value shown in the example is accurate. If the RMID and FMID values are equal, the *bbbbbbb* value need not be specified.
2. If the element entry in your target zone has any UMID values, you should first check to make sure the USERMOD itself was constructed so it works correctly in that environment. You should then make sure each of the UMID values is specified in the PRE operand in place of the *ccccccc* values shown in the example. This is not an absolute requirement, but if it is omitted, SMP/E issues warning messages during installation identifying that these SYSMODs may have an intersection with the one you are installing and, therefore, may be regressed. Putting the UMID values in the PRE list suppresses these messages.
3. If you do not specify the requisites that previously replaced an element, SMP/E does not allow your USERMODs to be installed unless you specify BYPASS(ID) on the APPLY or ACCEPT commands.
4. If you want this SYSMOD installed without any warning or error messages, you must specify all the current UMID values of each element in the SYSMOD in the PRE operand. This indicates to SMP/E that the SYSMOD was designed to be installed on the current function and service level of the element, including update level.

   If you do not specify the requisites that previously updated an element, the following occurs:

   - If your SYSMOD contains an element replacement, SMP/E does not allow the SYSMOD to be installed unless BYPASS(ID) is specified on the APPLY and ACCEPT command.
   - If your SYSMOD contained an element update, SMP/E allows the SYSMOD to be installed, but issues a warning message for each requisite not specified in the PRE list.

   This means SMP/E is unable to determine if there is an intersection between your update and those already on the element. It assumes that there is none, and you should investigate both updates to verify this.

Relationships to Other SYSMODs: Your SYSMOD may depend on another USERMOD or IBM PTF being installed, because you depend on the function provided by that SYSMOD. You may want to indicate that this SYSMOD is part of a set of USERMODs designed to provide some user function. Because each SYSMOD contains only one element, you want to tell SMP/E that they should all be installed together. This is done with the REQ operand (the *ccccccc* values in the example).

Specifying Superseded SYSMODs

If this SYSMOD is to fix multiple problems, each of the additional APARs that are being fixed should be specified in the SUP operand (*ddddddd* values in the example). This notifies SMP/E that it is not necessary to install those SYSMODs after the current SYSMOD is installed.

The ++JCLIN MCS

The ++JCLIN statement is necessary in all SYSMODs that add new modules or change the link-edit characteristics or link-edit control cards for existing load modules. The data following the ++JCLIN statement consists of the jobs necessary

for installing the new module or link-editing the affected load modules, as if that JCL were actually going to build the load modules from scratch from members of libraries.

The general format of the ++JCLIN statement is:

```
++JCLIN                    /* Installation data       */.
```

++MOD and ++ZAP MCSs

The ++MOD and ++ZAP statements are used to identify a replacement and update to a module in the distribution library and associated load modules.

The data following the ++MOD statement is an object deck. The general format of the ++MOD statement is:

```
++MOD(modname)             /* Distribution name       */
     DISTLIB(dlibname)     /* DLIB ddname             */
...
... Object deck for module
...
```

The data following the ++ZAP statement is a set of superzap control statements. The general format of the ++ZAP statement is:

```
++ZAP(modname)             /* Distribution name       */
     DISTLIB(dlibname)     /* DLIB ddname             */.
...
... Superzap control statement
...
```

The superzap control statements are the same as you would use if you were calling the superzap utility. The only exception is that on the NAME statement, you should specify only the CSECT name within the distribution library module, rather than the load module name and the CSECT name. When SMP/E installs the SYSMOD, it determines all the load modules into which this distribution module was link-edited and then calls the superzap utility for each of these load modules, modifying the NAME statement as appropriate.

++MAC and ++MACUPD MCSs

The ++MAC and ++MACUPD statements are used to identify a replacement and update to a macro in the distribution library and associated target libraries.

The data following the ++MAC statement is the macro replacement. The general format of the ++MAC statement is:

```
++MAC(macname)             /* Macro name              */
     DISTLIB(dlibname)     /* DLIB ddname             */.
...
... Macro replacement
...
```

The data following the ++MACUPD statement is the control statements that would have been used if you called the IEBUPDTE utility. The general format of the ++MACUPD statement is:

```
++MACUPD(macname)          /* Macro name              */
     DISTLIB(dlibname)     /* DLIB ddname             */.
...
... Update control statements
...
```

The following restrictions are enforced by SMP/E:

1. The first statement must be the "./ *change name=macname*" control statement.
2. The name specified on the change statement must be the same as on the ++MACUPD statement.
3. No insert or delete statement can be used. Inserting must be done by manually assigning each line a number. Deleting must be done by commenting out the line. This restriction enables SMP/E to merge the update control statement when multiple SYSMODs modify the same macro.

++SRC and ++SRCUPD MCSs

The ++SRC and ++SRCUPD statements are used to identify a replacement and update to source in the distribution library and associated target libraries. The format of the MCSs and the data format and restrictions are similar to those for macros.

The ++PROGRAM MCS

The ++PROGRAM statement is used to define replacements for program elements, which are pre-built load modules or program objects. (For a complete description of program elements, see the chapter on MCSs in *SMP/E Reference*.) You can add a program element by packaging it in a USERMOD.

The ++PROGRAM MCS is immediately followed by the load module or program object. The general format of the ++PROGRAM MCS is:

```
++PROGRAM(name)            /* Data element type, name */
     DISTLIB(dlibname)     /* DLIB ddname             */.
...
... Program element replacement
...
```

To be packaged inline, a program element must be unloaded along with its aliases into a sequential data set and then transformed with GIMDTS into the required fixed-block-80 record format before it is packaged. Later, when SMP/E installs the element, it is changed back to its original format. For more information about using GIMDTS, see *SMP/E Reference*.

Data Element MCSs

Data element MCSs are used to define replacements for data elements, which are elements other than macros, modules, program elements, and source code. These include TSO CLISTs, help panels, ISPF dialog panels, and online publications libraries. (For a complete description of data elements, see SMP/E Modification Control Statements in *SMP/E Reference*.) You can add a data element by packaging it in a USERMOD.

The data element MCS is immediately followed by the data element itself. The general format of the data element MCS is:

```
++element(name)            /* Data element type, name */
     DISTLIB(dlibname)     /* DLIB ddname             */.
...
... Data element replacement
...
```

To be packaged inline, a data element must contain fixed-block 80 records. If the original format of the element is not fixed-block 80 records, you can use GIMDTS, a service routine provided with SMP/E, to transform the element into the required

format before packaging it. Later, when SMP/E installs the element, it is reconverted to its original format. For more information about using GIMDTS, see *SMP/E Reference*.

Hierarchical File System Element MCSs

The hierarchical file system element MCSs are used to define a replacement for an element residing in a hierarchical file system (HFS). You can add a hierarchical file system element by packaging it in a USERMOD.

The hierarchical file system element MCS is immediately followed by the hierarchical file system element itself. The general format of a hierarchical file system element MCS is:

```
++hfs_element(hfsname)      /* HFS  name              */
     DISTLIB(dlibname)      /* DLIB ddname            */.
...
... Hierarchical file system element replacement
...
```

To be packaged inline, a hierarchical file system element must contain fixed-block 80 records. If the original format of the element is not fixed-block 80 records, you can use GIMDTS, a service routine provided with SMP/E, to transform the element into the required format before packaging it. Later, when SMP/E installs the element, it is reconverted to its original format. For more information about using GIMDTS, see *SMP/E Reference*.

Examples of USERMODs

This section contains examples of USERMODs that:
- Update a module
- Replace a module
- Add new modules
- Replace a macro or source code
- Update a macro or source code
- Add new source code
- Add source code that uses an IBM-supplied macro
- Add a module that uses an IBM-supplied macro

Example 1: Updating a Module

This is an example of a USERMOD that updates module MODULEA.

```
++USERMOD(USMP001)         /* SYSMOD ID of USERMOD.    */.
++VER(Z038)                 /* SREL for MVS.            */
     FMID(HMP1E00)          /* Applicable to SMPE.      */
     .                      /*                          */
++ZAP(MODULEA)             /* Name of module.          */
     DISTLIB(AOS12)         /* ddname of DLIB.          */
                            /*                          */.
  NAME MODULEA
* Verify existing ddname.
  VER  0050  404040404040404000
* Verify existing SYSOUT class.
  VER  0058  40
* Verify existing terminal assignment.
  VER  0059  00
* Replace with new data.
*            M Y P R I N T  = ddname
  REP  0050  D4E8D7D9C9D5E3
*            A              = SYSOUT class
  REP  0058  C1
```

Notes:

1. You should verify each location that you are going to update.
2. You can specify only one name in the NAME statement.
3. The changes made by the REP statements are explained by the preceding comment lines.

Example 2: Replacing a Module

This is an example of a USERMOD that replaces module MODULEB. After writing and assembling MODULEB, package it in a USERMOD, as shown below:

```
++USERMOD(USMP002)         /* SYSMOD is for USERMOD.   */.
++VER(Z038)                 /* SREL for MVS.            */
     FMID(HMP1E00)          /* Applicable to SMPE.      */
     .                      /*                          */
++MOD(MODULEB)             /* Name of module.          */
     DISTLIB(AOS12)         /* ddname of DLIB.          */
                            /*                          */.
...
... object module for MODULEB
...
```

Note: There are no PRE operands on the ++VER statement, because this module has not been modified since its initial installation. Therefore, only the ++VER FMID operand is required.

Example 3: Adding New Modules

This is an example of a USERMOD that adds new modules to create a new load module. In this example, the new modules are SMPMOD01, SMPMOD02, and SMPMOD03. These modules are link-edited to create load module SMPEXT01, whose entry point is SMPMOD01. The target library for SMPEXT01 is SYS1.LINKLIB. The following example shows how the new modules and load modules can be packaged in a USERMOD.

```
++USERMOD(USMP003)         /* SYSMOD ID of USERMOD.    */.
++VER(Z038)                 /* SREL for MVS.            */
     FMID(HMP1E00)          /* Applicable to SMPE.      */
                            /*                          */.
++JCLIN                     /* JCLIN to install routine.*/.
//JOB1     JOB 'accounting info',MSGLEVEL=(1,1)
//STEP1    EXEC PGM=IEWL
//PVTDLIB1 DD DSN=SYS1.PVTDLIB1,DISP=OLD
//SYSLMOD  DD DSN=SYS1.LINKLIB,DISP=OLD
//SYSPRINT DD SYSOUT=A
//SYSLIN   DD *
  INCLUDE  PVTDLIB1(SMPMOD01)
  INCLUDE  PVTDLIB1(SMPMOD02,SMPMOD03)
  ENTRY    SMPMOD01
  NAME     SMPEXT01(R)
/*
++MOD(SMPMOD01)            /* Customized exit routine. */
     DISTLIB(PVTDLIB1)      /* ddname of DLIB.          */
                            /*                          */.
...
... object module for SMPMOD01
...
++MOD(SMPMOD02)            /* Customized exit routine. */
     DISTLIB(PVTDLIB1)      /* ddname of DLIB.          */
                            /*                          */.
...
... object module for SMPMOD02
...
++MOD(SMPMOD03)            /* Customized exit routine. */
     DISTLIB(PVTDLIB1)      /* ddname of DLIB.          */
                            /*                          */.
...
... object module for SMPMOD03
...
```

Notes:

1. The ++JCLIN statement is required because the SYSMOD provides new modules and a new load module.
2. This SYSMOD could have been packaged as a ++FUNCTION, because each of the modules is user-written.
3. This SYSMOD could have been packaged as three separate SYSMODs, each containing one module. In that case, each SYSMOD would then need to specify the ++VER REQ operand so the SYSMODs would be installed as a set.

Example 4: Replacing a Macro or Source Code

This is an example of a USERMOD that adds a new macro. In this example, the macro is installed in a target and a distribution library. In addition, source code element USRSRC01 is to be assembled when the macro is installed.

```
++USERMOD(UJES004)         /* SYSMOD ID of USERMOD.    */.
++VER(Z038)                /* SREL for MVS.            */
     FMID(HJE2102)         /* Applicable to JES.       */
                           /*                          */.
++MAC(USRMAC01)            /* Name of new macro.       */
     DISTLIB(USRMACLB)     /* ddname of DLIB.          */
     SYSLIB (MACLIB)       /* ddname of system MACLIB. */
     ASSEM  (USRSRC01)     /* Reassemble this source.  */
                           /*                          */.
...
... macro replacement
...
```

Notes:

1. No JCLIN is required to install a new macro; all necessary information can be specified on the ++MAC statement.
2. You can follow this example to add new source code, with the following exceptions:
   - The ASSEM operand is not required for source code. If SMP/E understands where source code is to be installed, it automatically tries to assemble the new version of the source code after replacing the old version.
   - To define how the source code should be installed, you should include a ++JCLIN statement followed by JCLIN data. The JCLIN data should be similar to that in the example of adding a new module.

Example 5: Updating a Macro or Source Code

The following is an example of a USERMOD to update a macro that was added by a previous USERMOD.

```
++USERMOD(UJES005)          /* SYSMOD ID of USERMOD.    */.
++VER(Z038)                 /* SREL for MVS.            */
     FMID(HJE2102)          /* Applicable to JES.       */
     PRE (UJES004)          /* Previous replacement.    */
                            /*                          */.
++MACUPD(USRMAC01)          /* Update customized macro. */
     DISTLIB(USRMACLB)      /* ddname of DLIB.          */
                            /* SMPE knows SYSLIB.       */
     ASSEM(USRSRC01)        /* Reassemble this source.  */
                            /*                          */.
./ CHANGE NAME=USRMAC01
...
... macro changed lines here with
... sequence numbers in columns 73–80
...
```

Notes:

1. The ++VER PRE operand specifies the previous SYSMOD that replaced the macro.
2. You can follow this example to update source code, with the following exceptions:
   - The ASSEM operand is not used for source code.
   - Because the SYSMOD adding the source code defined how the module should be installed, there is no need to repeat that information on a ++JCLIN statement in the USERMOD.

Example 6: Adding New Source Code

Assume that you have written new source code to be added to an existing product. It is a member in one of your partitioned data sets. To be installed, it must be assembled, then copied as a single-module load module into its target library. Table 15 shows the information you need to specify and where to specify it.

Table 15. Information Needed to Add New Source Code

Information to Provide:	Where to Specify It:
1 Name of source code element: IFBSRC01	• Element name on ++SRC MCS: **++SRC(IFBSRC01)**

Table 15. Information Needed to Add New Source Code (continued)

Information to Provide:	Where to Specify It:
2 Name of object module produced by assembly: IFBSRC01 (same as source code element)	• JCLIN data, M= value on COPY SELECT statement: `COPY SELECT M=(IFBSRC01)`

3 Where input source code is provided:
PDS member
	• TXLIB value on ++SRC MCS:
	`++SRC(IFBSRC01) TXLIB(REPLACE)`
	• DDDEF entry or DD statement for APPLY and ACCEPT processing:
	`REPLACE DD DSN=...`

4 Target library for source:
SYS1.IFBSRC
	• SYSLIB value on ++SRC MCS:
	`++SRC(IFBSRC01) SYSLIB(IFBSRC)`
	• JCLIN data, OUTDD value on COPY statement:
	`COPY INDD=inddval,OUTDD=IFBSRC TYPE=SRC`
	• DDDEF entry or DD statement for APPLY processing:
	`IFBSRC DD DSN=SYS1.IFBSRC`

5 Target library for load module:
SYS1.LPALIB
	• JCLIN data, OUTDD value on COPY statement:
	`COPY INDD=inddval,OUTDD=LPALIB TYPE=MOD`
	• DD statement in JCLIN data and in SMP/E job for APPLY processing:
	`LPALIB DD DSN=SYS1.LPALIB`

6 Distribution library for source code:
SYS1.AIFBSRC
	• DISTLIB value on ++SRC MCS:
	`++SRC(IFBSRC01) DISTLIB(AIFBSRC)`
	• JCLIN data, INDD value on COPY statement:
	`COPY INDD=AIFBSRC,OUTDD=IFBSRC TYPE=SRC`
	• DD statement or DDDEF entry for ACCEPT processing:
	`AIFBSRC DD DSN=SYS1.AIFBSRC`

7 Distribution library for assembled object module:
SYS1.AOS23
	• DISTMOD value on ++SRC MCS:
	`++SRC(IFBSRC01) DISTMOD(AOS23)`
	• JCLIN data, INDD value on COPY statement:
	`COPY INDD=AOS23,OUTDD=LPALIB  TYPE=MOD`
	• DD statement in JCLIN data and in SMP/E job during APPLY and ACCEPT processing:
	`AOS23 DD DSN=SYS1.AOS23`

Table 15. Information Needed to Add New Source Code (continued)

Information to Provide:	Where to Specify It:

8 How to install the source code:
Assemble, then copy
	• ++SRC MCS automatically notifies SMP/E to assemble the module.
	• JCLIN data, COPY steps:
	`//JOB1 JOB ...`
	`//STEP1 EXEC PGM=IEBCOPY`
	`//AIFBSRC DD DSN=SYS1.AIFBSRC,...`
	`//IFBSRC DD DSN=SYS1.IFBSRC,...`
	`//AOS23 DD DSN=SYS1.AOS23,...`
	`//LPALIB DD DSN=SYS1.LPALIB,...`
	`//SYSIN DD *`
	`COPY INDD=AIFBSRC,OUTDD=IFBSRC TYPE=SRC`
	`SELECT M=(IFBSRC01)`
	`COPY INDD=AOS23,OUTDD=LPALIB TYPE=MOD`
	`SELECT M=(IFBSRC01)`
	`/*`
	• The OPTIONS and UTILITY entries in effect for the APPLY or ACCEPT command being processed specify the names of the assembler and copy utilities to use and parameters to be passed to them.

The following is a sample USERMOD that can be used to package the source code. The numbers associate items in the SYSMOD with the information listed in Table 15 on page 155.

```
                ++USERMOD(USR0001)          /* My USERMOD.         */.
                ++VER(Z038) FMID(MYFMID1).  /* For my function.    */.
  8             ++JCLIN.                    /* Link object module. */.
                //JOB1    JOB ...
                //STEP1   EXEC PGM=IEBCOPY
  6             //AIFBSRC DD DSN=SYS1.AIFBSRC,DISP=SHR
  4             //IFBSRC  DD DSN=SYS1.IFBSRC,DISP=SHR
  7             //AOS23   DD DSN=SYS1.AOS23,DISP=SHR
  5             //LPALIB  DD DSN=SYS1.LPALIB,DISP=SHR
                //SYSIN   DD *
  4 , 6           COPY INDD=AIFBSRC,OUTDD=IFBSRC  TYPE=SRC
  2               SELECT M=(IFBSRC01)
  5 , 7           COPY INDD=AOS23,OUTDD=LPALIB  TYPE=MOD
  2               SELECT M=(IFBSRC01)
                /*
1 , 2 , 8       ++SRC(IFBSRC01)             /* My source module.   */
  3                 TXLIB(REPLACE)          /* Where source is.    */
  6                 DISTLIB(AIFBSRC)        /* DISTLIB for source. */
  7                 DISTMOD(AOS23)          /* DISTLIB for object. */
  4                 SYSLIB(IFBSRC)          /* SYSLIB for source.  */.
```

Example 7: Adding New Source Code that Uses an IBM-Supplied Macro

Assume you are packaging one of your user-written routines as a USERMOD (UMOD001). Your USERMOD includes assembler source (SRCPART), which is to be included in load module LOADMOD1 and is packaged as a SRC element with a ++SRC statement, as described in previous USERMOD examples. SRCPART refers to an IBM-supplied macro (IBMMAC), which was packaged in its owning product with a ++MAC statement. You want SRCPART to be automatically reassembled every time IBMMAC is updated or replaced with service. To accomplish this, your USERMOD must contain a JCLIN assembly step in addition to the necessary SMP/E MCS statements. (This means you need to supply the

same SRCPART definition as part of the JCLIN input, as well as after the ++SRC statement.) Your USERMOD might look something like this:

```
++USERMOD(UMOD001).
++VER(srel) FMID(fmid).
++JCLIN.
//STEP1    EXEC PGM=IEUASM
//SYSPUNCH DD DSN=&&PUNCH(SRCPART),DISP=(PASS);
//SYSIN    DD *
SRCPART CSECT
        IBMMAC
        BR   14
        END
//LINK     EXEC PGM=IEWL
//SYSLMOD  DD DSN=SYS1.LINKLIB,DISP=SHR
//SYSLIN   DD *
  INCLUDE  SYSPUNCH(SRCPART)
  NAME LOADMOD1
/*
++SRC(SRCPART) SYSLIB(tgtlib) DISTLIB(dlib).
SRCPART CSECT
        IBMMAC
        BR   14
        END
```

When the assembly step in the JCLIN is processed, a GENASM subentry is added to the MAC entry for IBMMAC indicating SRCPART is to be assembled whenever IBMMAC is updated. So, if you install a SYSMOD that updates IBMMAC, SMP/E automatically reassembles SRCPART and link-edits the new copy into LOADMOD1.

Note: Remember, when a link-edit step contains a SYSLMOD DD statement, SMP/E uses the low-level qualifier of the data set name to determine the ddname of the load module's target library (and, by extension, the name of the DDDEF entry to use when allocating this library). In this USERMOD, for example, SMP/E looks for a DDDEF entry named LINKLIB in order to allocate the SYSLMOD data set.

Example 8: Adding a New Module that Uses an IBM-Supplied Macro

Assume you are packaging one of your user-written routines as a USERMOD (UMOD001). Your USERMOD includes an object module (SRCPART), which is to be included in load module LOADMOD1 and is packaged as a MOD element with a ++MOD statement, as described in previous USERMOD examples. SRCPART refers to an IBM-supplied macro (IBMMAC), which was packaged in its owning product with a ++MAC statement. You want to be notified whenever IBMMAC is updated or replaced with service so you can update your module and reapply your USERMOD. To accomplish this, your USERMOD must include the macro, and must specify the last SYSMOD that replaced the macro (the RMID value in the MAC entry) and all the SYSMODs that have updated the macro (the UMID values in the MAC entry) as prerequisites. Your USERMOD might look something like this:

```
++USERMOD(UMOD001).
++VER(srel) FMID(fmid) PRE(macrmid).
++JCLIN.
//LINK     EXEC PGM=IEWL
//SYSLMOD  DD DSN=SYS1.LINKLIB,DISP=SHR
//SYSLIN   DD *
  INCLUDE  DLIB1(SRCPART)
  NAME LOADMOD1
/*
++MAC(IBMMAC).
  ...
  macro source
  ...
++MOD(SRCPART) DISTLIB(dlib).
  ...
  module object deck
  ...
```

Because you specified the macro's RMID and UMIDs, when IBMMAC is serviced, the APPLY will get a MODID error for the USERMOD. You will have to restore the USERMOD to successfully apply the service. Then you can rework the USERMOD and apply it again.

Note: Remember, when a link-edit step contains a SYSLMOD DD statement, SMP/E uses the low-level qualifier of the data set name to determine the ddname of the load module's target library (and, by extension, the name of the DDDEF entry to use when allocating this library). In this USERMOD, for example, SMP/E looks for a DDDEF entry named LINKLIB in order to allocate the SYSLMOD data set.

Chapter 17. Determining Which SYSMODs Led Others to Fail: The Causer SYSMOD Summary Report

This chapter describes root cause analysis and provides examples of how to use the causer SYSMOD information in the Causer SYSMOD Summary report and the SYSMOD Status report to determine the cause of SYSMOD failure.

Introduction

A *causer SYSMOD* is a SYSMOD whose failure led to the failure of other SYSMODs. To identify causer SYSMODs, SMP/E performs root cause analysis for the ACCEPT, APPLY, and RESTORE commands. The types of errors SMP/E analyzes to determine causer SYSMODs include the following:

- Held SYSMODs
- Missing requisite SYSMODs
- Utility program failures: copy, update, assembler, link, zap
- Out-of-space conditions: *x*37 ABENDs
- Missing DD statements and other allocation errors
- ID errors (a SYSMOD does not supersede or specify as a prerequisite an RMID or a UMID of an element)
- JCLIN errors (syntax errors)

The results of SMP/E's root cause analysis are presented in two reports:

- SYSMOD Status report

  This report summarizes the processing done for each eligible SYSMOD. For SYSMODs that failed, the report lists the causer SYSMODs.

  After you check the SYSMOD Status report to determine the results of processing, use the Causer SYSMOD Summary report to determine which errors need to be corrected.
- Causer SYSMOD Summary report

  This report lists the causer SYSMODs along with a brief summary of the related messages, descriptions of the errors that caused the SYSMODs to fail, and, when feasible, some causes for those errors.

Using Causer SYSMOD Information

How you use the causer SYSMOD information provided by the Causer SYSMOD Summary report and the SYSMOD Status report depends on what you need to install—all the SYSMODs specified on the command you are processing, or only specific ones. This section describes some general scenarios and includes an example of using these reports.

Resolving Errors for All SYSMODs That Failed

Suppose you are installing a CBPDO, but the reports show that several of the SYSMODs failed. You want to resolve all the reported errors and install all the SYSMODs in the CBPDO.

1. Go to the Causer SYSMOD Summary report to identify the causer SYSMODs and determine how to resolve the errors.

2. Resolve the errors.
3. Rerun the command that failed. If you find more errors on this pass, repeat the process.

Resolving Errors for a Single SYSMOD That Failed

Suppose you are installing a group of SYSMODs, but the reports show that several of them have failed. You want to resolve the errors for one specific SYSMOD and install it.

1. Use the SYSMOD Status report to determine the causer SYSMOD for the SYSMOD that you need to install.
2. Go to the Causer SYSMOD Summary report and determine how to resolve the errors for the causer SYSMOD.
3. Resolve the errors.
4. Rerun the command to install the SYSMOD that previously failed. If you find more errors on this pass, repeat the process.

Example

Suppose you ran the following command:

```
APPLY  S(UR00001) GEXT CHECK.
```

and the SYSMOD Status report included the results shown in Figure 36.

```
Page nnnn  - NOW SET TO TARGET ZONE nnnnnnn  DATE mm/dd/yy  TIME hh:mm:ss SMP/E 32.nn SMPRPT OUTPUT


 SYSMOD STATUS REPORT FOR APPLY PROCESSING     SYSMODS APPLIED - 0

 NOTE:  '-' INDICATES THE REQUISITE SYSMOD OR HOLD CONDITION IS NOT SATISFIED
        '*' INDICATES THE NON SATISFIED REQUISITE SYSMOD OR HOLD CONDITION IS BYPASSED
        '#' INDICATES THE SUPERSEDING SYSMOD WAS NOT PROCESSED

 SYSMOD   STATUS    TYPE      FMID      REQUISITE SYSMODS, SUPBY SYSMODS, HOLD REASON-IDS, AND CAUSER SYSMODS

 AR00001  NOGO                          SUPBY  #UR00002

 UR00001  HELD      PTF       HROOTC1   HOLDE  -AR00001
                                        CAUSER  UR00002

 UR00002  HELD      PTF       HROOTC1   HOLDE  -AR00003
                                        CAUSER  UR00002
```

Figure 36. SYSMOD Status Report: Sample Report for APPLY

In this case, the report indicates that APPLY processing failed for SYSMODs UR00001 and UR00002 because of unresolved error holds. The causer SYSMOD for both PTFs is UR00002. Next, you look up UR00002 in the Causer SYSMOD Summary report, shown in Figure 37 on page 163.

```
Page nnnn  - NOW SET TO TARGET ZONE nnnnnnnn  DATE mm/dd/yy  TIME hh:mm:ss SMP/E 32.nn SMPRPT OUTPUT

CAUSER SYSMOD SUMMARY REPORT FOR APPLY PROCESSING

CAUSER     FMID     MESSAGE ID  PAGE    ERROR DESCRIPTION AND POSSIBLE CAUSES

UR00002    HROOTC1  GIM35901I      2    ERROR HOLD AR00003 WAS NOT RESOLVED.
```

Figure 37. Causer SYSMOD Summary Report: Sample Report for APPLY

That report shows that APPLY processing failed because error hold AR00003 was not resolved for SYSMOD UR00002. By resolving this hold, you will fix the errors listed in the SYSMOD Status report.

Chapter 18. Java Archive Update Exploiter's Guide

This chapter is a guide to packaging Java Archive files (JAR files) and updates for JAR files in SMP/E. The intended audience for this section is anyone who will be exploiting this function, including product packagers and service teams. This guide provides information and examples about packaging a product release using the ++JAR MCS and building subsequent PTFs to service that release using the ++JARUPD MCS.

JAR Replacements in FMIDs

Suppose your product has a simple TicTacToe applet. The complete TicTacToe applet (or Java application) is composed of a class file and audio and image files, and is stored in a directory structure as follows:

```
/u/pezk/TicTacToe/TicTacToe.class
                 /audio/beep.au
                       /ding.au
                       /yahoo.au
                 /images/cross.gif
                        /not.gif
```

To build a JAR file element for the complete TicTacToe applet, you would first make your working directory the TicTacToe directory where the applet files reside, and then run the following jar command:

```
cd /u/pezk/TicTacToe/
jar cvf ABCTTT.jar *
```

Specified in this way, the jar command takes all files in the working directory (/u/pezk/TicTacToe), and all files within subdirectories of the working directory, and creates a JAR file named ABCTTT.jar.

Your product release, or FMID, can then contain this single JAR file, which is the complete TicTacToe applet. You can do this using the ++JAR MCS as seen in the following example:

```
++JAR(ABCTTT) DISTLIB(AABCBIN) SYSLIB(SABCBIN) RELFILE(2)
  PARM(PATHMODE(0,6,4,4))
  LINK('../TicTacToe.jar')
  SYMLINK('../../../../../usr/lib/TicTacToe.jar')
  SYMPATH('../../usr/lpp/abc/bin/TicTacToe.jar').
```

JAR Updates in PTFs

Suppose some time after your FMID has been made available and is being used, a defect is discovered (an APAR). The TicTacToe applet requires a change because of the APAR. Specifically, you need to add another image file (images/new.gif), and replace the class file (TicTacToe.class) with an updated copy of the class file. Further, suppose the change team has placed the new image file and updated class file in directories, as follows:

```
/u/apars/ow12345/TicTacToe/TicTacToe.class
/images/new.gif
```

You can instruct SMP/E to add new files to a JAR file, and replace files in an existing JAR file by using the ++JARUPD MCS. If files TicTacToe.class and new.gif were to be packaged and archived together in a JAR file of their own, then that

file, in SMP/E terms, would be considered a JAR update file. For example, given the directory structure above for the new and updated files, the jar command could be used to create the JAR update file as follows:

```
cd /u/apars/ow12345/TicTacToe/
jar cvf ABCTTT.jarupd *
```

The resultant JAR file ABCTTT.jarupd would be packaged as a ++JARUPD in a PTF as in the following example:

```
++PTF(UW12345).
++VER(Z038) FMID(fmid).
++JARUPD(ABCTTT)
    PARM(PATHMODE(0,6,4,4)) JARPARM(0M)
    LINK('../TicTacToe.jar')
    SYMLINK('../../../../../usr/lib/TicTacToe.jar')
    SYMPATH('../../usr/lpp/abc/bin/TicTacToe.jar').
```

Suppose now another defect (APAR) is discovered against the TicTacToe applet, and you must update the /audio/beep.au file within the archive. Again, you can use the ++JARUPD MCS to describe an update to the archive. Assuming the replacement audio file resides in a directory as follows:

```
/u/apars/ow54321/TicTacToe/audio/beep.au
```

The following jar command could create the necessary JAR update:

```
cd /u/apars/ow54321/TicTacToe/
jar cvf ABCTTT.jarupd *
```

The resulting JAR file ABCTTT.jarupd would be packaged as a ++JARUPD in a PTF as in the following example:

```
++PTF(UW54321).
++VER(Z038) FMID(fmid) PRE(UW12345).
++JARUPD(ABCTTT)
    PARM(PATHMODE(0,6,4,4)) JARPARM(0M)
    LINK('../TicTacToe.jar')
    SYMLINK('../../../../../usr/lib/TicTacToe.jar')
    SYMPATH('../../usr/lpp/abc/bin/TicTacToe.jar').
```

Notice the second PTF has a prerequisite for the first PTF. Such a relationship is required by SMP/E because both PTFs update the same JAR file.

JAR Replacements in PTFs

Just as replacements for JAR files can be delivered in an FMID, so can JAR file replacements be delivered in PTFs. Suppose you decide to create a "level-set" PTF, which incorporates the updates from several previous APARs and PTFs. In this case, you want to replace the entire TicTacToe applet JAR file, rather then add or replace files within the archive.

You create the JAR file for the applet using the jar command as discussed earlier and then package it as a ++JAR element in a PTF, as in the following example:

```
++PTF(UW87654).
++VER(Z038) FMID(fmid) SUP(UW12345 UW54321).
++JAR(ABCTTT) DISTLIB(AABCBIN) SYSLIB(SABCBIN)
     PARM(PATHMODE(0,6,4,4)) JARPARM(0M)
     LINK('../TicTacToe.jar')
     SYMLINK('../../../../../usr/lib/TicTacToe.jar')
     SYMPATH('../../usr/lpp/abc/bin/TicTacToe.jar').
```

Notice this PTF supersedes both PTFs that supplied updates for the JAR file. Again, such relationships (either supersede or prerequisite) are required by SMP/E, because both PTFs provided updates for the JAR file contained in the current PTF.

Appendix A. Migration

Migration Overview

Your plan for migrating to the new level of SMP/E should include information from a variety of sources. These sources of information describe topics such as coexistence, service, migration procedures, and interface changes.

The following documentation, which is supplied with your product order, provides information about installing your z/OS system. In addition to specific information about SMP/E, this documentation contains information about all of the z/OS elements.

- *z/OS and z/OS.e Planning for Installation*

  This book describes the installation requirements for z/OS at a system and element level. It includes hardware, software, and service requirements for both the driving and target systems. It also describes any coexistence considerations and actions.
- *SMP/E Program Directory*

  This document, which is provided with your SMP/E product order, leads you through the specific installation steps for SMP/E.
- *ServerPac Installing Your Order*

  This is the order-customized installation book for using the ServerPac installation method. Be sure to review "Appendix A. Product Information", which describes data sets supplied, jobs or procedures that have been completed for you, and product status. IBM may have run jobs or made updates to PARMLIB or other system control data sets. These updates could affect your migration.

Within this book, you can find information about the specific updates and considerations that apply to this release of SMP/E.

- "Migration Roadmap" on page 171

  This section identifies the migration paths that are supported with the current level of SMP/E. It also describes the additional publications that can assist you with your migration to the current level.
- "SMP/E V3R2 Overview" on page 172

  This section describes the specific updates that were made to SMP/E for the current release. For each item, this section provides an overview of the change, a description of any migration and coexistence tasks that may be considered, and where you can find more detailed information in the SMP/E library or other element libraries.
- "Summary of Interface Changes" on page 188

  This section provides a summary of the changes that are made to z/OS user and programming interfaces.

Terms you need to know

This section describes some terms you may need to know as you use this section.

Migration Activities that relate to the installation of a new version or release of a program to replace an earlier level. Completion of these

activities ensures that the applications and resources on your system will function correctly at the new level.

Coexistence Two or more systems at different levels (for example, software, service, or operational levels) that share resources. Coexistence includes the ability of a system to respond in the following ways to a new function that was introduced on another system with which it shares resources: ignore a new function, terminate gracefully, support a new function.

New releases of SMP/E generally add to or enhance the information stored in SMP/E data sets. This new or enhanced data is not always compatible with previous releases of SMP/E. Results are unpredictable when a prior release of SMP/E encounters data introduced by a later release. Therefore, IBM provides toleration PTFs that allow selected prior releases to ignore changes introduced by a later release. In some cases, a compatibility PTF or small programming enhancement (SPE) is available to add a new function to selected prior releases. *z/OS and z/OS.e Planning for Installation*, GA22-7504, provides a list of the required toleration and compatibility PTFs for each prior release of SMP/E.

SMP/E Release Levels

The SMP/E element of z/OS is usually not updated with every release of z/OS. For example, the level of SMP/E that was shipped with z/OS Release 1 is identical to that shipped with OS/390 V2R7.

Developing a migration strategy

The recommended steps for migrating to a new release of SMP/E are:

1. Become familiar with the supporting migration and installation documentation for the release.

   You should determine what updates are needed for products that are supplied by IBM, system libraries, and non-IBM products. Review *z/OS and z/OS.e Planning for Installation* and *z/OS Introduction and Release Guide* for information about SMP/E and other z/OS elements.
2. Develop a migration plan for your installation.

   When planning to migrate to a new release of SMP/E, you must consider high-level support requirements, such as machine and programming restrictions, migration paths, and program compatibility.
3. Obtain and install any required program temporary fixes (PTFs) or updated versions of the operating system.

   Call the IBM Software Support Center to obtain the preventive service planning (PSP) upgrade for SMP/E, which provides the most current information about PTFs for SMP/E. Check RETAIN again just before testing SMP/E. For information about how to request the PSP upgrade, refer to the *SMP/E Program Directory*. Although the *SMP/E Program Directory* contains a list of the required PTFs, the most current information is available from the IBM Software Support Center.
4. Install the product using the *SMP/E Program Directory* or the *ServerPac Installing Your Order* documentation.
5. Verify that any application programs that use the SMP/E API will continue to run and, if necessary, make changes to ensure compatibility with the new release.
6. Use the new release before initializing major new function.

7. If necessary, customize the new function for your installation.
8. Exercise the new functions.

Reviewing changes to SMP/E processing

As you define your installation's migration plan, consider how the new and changed SMP/E functions might affect the following areas of SMP/E processing. for each item described in "SMP/E V3R2 Overview" on page 172, you should review the "Migration procedures" section to determine how, or if, the change affects the tasks that are performed at your installation.

Customization
You can customize some SMP/E functions to take advantage of new support after the product is installed. For example, you can tailor SMP/E through the use of installation exit routines or SMPPARM options. This section lists changes to SMP/E that might require you to tailor SMP/E to ensure that it runs as before.

Operations
The new SMP/E release might introduce changes to its operating characteristics, such as changed commands, new or changed messages, or in the methods of implementing new functions. This book identifies those changes for which you should provide user education before using this release of SMP/E.

Programming
To ensure that existing programs run as before, programmers who develop programs that use the GIMAPI, library change file records, or UNIX shell scripts must be aware of any changes to those interfaces introduced in a new release of SMP/E.

Reviewing changes to SMP/E interfaces

When defining your installation's migration plan, also consider that SMP/E interfaces may also be affected by the new or changed functions that are introduced in this release. These interfaces include:

- Commands
- Exits
- Macros
- Messages
- Panels
- SYS1.SAMPLIB members

"Summary of Interface Changes" on page 188 provides a summary of the changes that affect these interfaces for the release. This information is also listed in the "What this change affects" section that is provided for each release enhancement in "SMP/E V3R2 Overview" on page 172.

Migration Roadmap

SMP/E V3R2 Summary

Table 16 on page 172 summarizes the updates that have been introduced to SMP/E V3R2. If you are migrating from z/OS V1R1 or OS/390, you should review the information in the detailed section for each item.

Table 16. Summary of SMP/E V3R2 Updates

For Information About:	Refer to:
"SMP/E Dialog Customization" on page 176	176
"Java Archive Files" on page 174	174
"Smaller SMPLTS data set" on page 174	174
"DUMMY data set for SYSDEFSD" on page 175	175
"GIMZIP: Archive Segmentation" on page 173	173
"GIMZIP: User Defined Subdirectories" on page 173	173
"LINK LMODS Command"	172
"UPGRADE Command"	172
"REPORT CALLLIBS Command Removal"	172
"GIMXSID Service Routine" on page 173	173
"GIMUTTBL Removal" on page 176	176

SMP/E V3R2 Overview

The following sections describe the new and changed SMP/E functions that are introduced for SMP/E V3R2. The information about each item includes:

- Description
- Summary of the SMP/E tasks or interfaces that might be affected
- Coexistence considerations, if any, that are associated with the item
- Migration procedures, if any, that are associated with the item
- References to other publications that contain additional detailed information

LINK LMODS Command

The new LINK LMODS command can be used to refresh the callable services for all load modules within a particular target zone. The LINK LMODS command can also be used to refresh the callable services only for those load modules that have a dependency on a particular set of CALLLIBS. The LINK LMODS command replaces the REPORT CALLLIBS command, which has been removed from SMP/E V3R2. For more information about the LINK LMODS command, refer to *SMP/E Commands*.

REPORT CALLLIBS Command Removal

The REPORT CALLLIBS command has been removed from SMP/E V3R2. It has been replaced by the LINK LMODS command.

UPGRADE Command

New releases of SMP/E must sometimes make changes to SMP/E data sets that cannot be properly processed by prior SMP/E releases. SMP/E usually makes incompatible changes only when necessary to provide new and improved capabilities. For example, a new type of element requires a new entry type in SMPCSI data sets and these new entry types are typically not understood or processed correctly by SMP/E levels that have not been specifically updated to do so.

The UPGRADE command allows you to specify when SMP/E is permitted to make incompatible changes to SMP/E data sets. This, in turn, allows you to make

the trade-off between exploiting new SMP/E functions and preserving compatibility with prior SMP/E releases. For more information about the UPGRADE command, refer to *SMP/E Commands*.

Coexistence considerations

A toleration PTF will enable OS/390 V2R7 SMP/E, z/OS V1R2 SMP/E, and SMP/E V3R1 to automatically check for incompatible changes made by a higher level of SMP/E. A toleration PTF will also provide a warning message should a user try to issue an UPGRADE command on a release of SMP/E prior to V3R2.

GIMXSID Service Routine

The GIMXSID service routine is used as part of the ShopzSeries offering. GIMXSID creates a single data source required by ShopzSeries to place customized software product and service orders. The data source created by GIMXSID, the software inventory data, is a composite of three kinds of information as follows:

1. A list of FEATUREs found in the SMPCSI data sets. The Feature List is used by ShopzSeries to perform product requisite checking and also to prime the order checklist when ordering a ServerPac.
2. A list of the FMIDs found in the SMPCSI data sets. The FMID List is used by ShopzSeries to scope service orders to the PTFs applicable solely to the user's desired configuration of target and global zones.
3. A bitmap representation of the PTFs found in the specified target zones and global zones. The PTF Bitmap is used by ShopzSeries (CCSS) to produce service packages that do not contain PTFs that are already present in the user's configuration.

GIMZIP: Archive Segmentation

The GIMZIP service routine has been enhanced with a new SEGMENT option to allow you to specify the maximum size for the network transportable objects produced by GIMZIP. Very large objects will be divided into archive segments that are within the specified size. The GIMUNZIP service routine and the RECEIVE FROMNETWORK and RECEIVE FROMNTS commands will now accept network packages that contain archive segments as input. For more information about the GIMZIP and GIMUNZIP service routines, refer to *SMP/E Reference*. For more information about the RECEIVE commands, refer to *SMP/E Commands*.

Also, you can now use the SMPWKDIR DD statement to specify a location for temporary files produced during GIMZIP processing.

Coexistence considerations

SMP/E releases prior to SMP/E V3R2 cannot process GIMZIP output that contains segmented archive files. A toleration PTF will issue an error message should a user try to process GIMZIP output that contains segmented archive files on a release of SMP/E prior to V3R2.

GIMZIP: User Defined Subdirectories

Users of GIMZIP may now specify subdirectories in which to store documentation, samples, readme files, and other files. This is done with the new **subdir** attribute of the <FILEDEF> tag of the GIMZIP package control statement. The subdirectory is created in the hierarchical file system, within the parent directory pointed to by the SMPDIR DD statement in the JCL used to invoke GIMZIP.

Coexistence considerations

Unless the required PTF is installed, SMP/E releases prior to SMP/E V3R2 cannot process GIMZIP packages that exploit user-defined subdirectories.

Java Archive Files

Many z/OS software products developed with Java use Java Archive (JAR) files as the packaging format for Java application files. To better accommodate such products, SMP/E V3R2 is introducing JAR file update support. For SMP/E, there will be two forms of JAR files; JAR replacement files and JAR update files. JAR replacement files are complete replacements for a JAR file and are treated simply as files in the hierarchical file system. SMP/E will copy replacement JAR files into the hierarchical file system, just as is done for all other hierarchical file system elements. JAR update files are archive files themselves, but do not contain all of the component files that make up the complete JAR file. A JAR update file contains only the new and changed component files. These new and changed component files are archived into a JAR file. SMP/E uses the JAR command and the contents of a JAR update file to update an existing JAR file.

For information about the JAR command and JAR files, refer to the "Java Developer Connection, Using JAR Files: The Basics", at http://developer.java.sun.com/developer/Books/JAR/basics/index.html

Coexistence considerations

SMP/E releases prior to SMP/E V3R2 cannot process JAR replacement files or JAR update files, nor can they process the data entries for these elements. A toleration PTF will cause SMP/E to issue an error message if it encounters an unsupported JAR element or data entry.

Smaller SMPLTS data set

The SMPLTS data set is used by SMP/E to save the base version of a product's load modules that use callable services. To reduce SMPLTS space requirements, SMP/E now saves a base version of a load module in the SMPLTS data set only if it contains both CALLLIBS and XZMOD subentries. If a load module contains CALLLIBS subentries, but no XZMOD subentries, this load module is not saved in the SMPLTS.

For more detailed information about these command changes, refer to *SMP/E Commands*.

Coexistence considerations

The SMPLTS data set used by SMP/E V3R2 may not contain the base version of load modules with CALLLIBS subentries. Because of this, once the SMPLTS has been modified by SMP/E V3R2, you cannot use certain commands from older levels of SMP/E that depend on the base version of a load module being in the SMPLTS data set. Specifically, if any of the following updates were made to a zone with load modules containing CALLLIBS but no XZMOD subentries, the target zone is marked:

- CLEANUP command is run against the zone
- GENERATE command is run against the zone
- APPLY or RESTORE command is run and the base version of the load module is deleted from the SMPLTS

A toleration PTF against older releases of SMP/E allows certain commands to be processed against a target zone that has been marked to indicate that the SMPLTS data set can not be used. These commands do not depend on the base version of load modules existing in the SMPLTS:

- APPLY
- BUILDMCS
- CLEANUP
- DEBUG
- GENERATE
- JCLIN
- LIST
- LOG
- RESETRC
- SET
- UCLIN
- UNLOAD
- ZONECOPY
- ZONEDELETE
- ZONEEDIT
- ZONEEXPORT
- ZONEIMPORT
- ZONEMERGE

The toleration PTF also prohibits the RESTORE or LINK MODULE command from being run against a target zone that has been marked.

DUMMY data set for SYSDEFSD

The SYSDEFSD DD statement defines the location for the binder to write IMPORT statements for all exported entries of a DLL load module. Whenever entries are to be exported, the binder expects the SYSDEFSD data set and, if it is not present, will issue a warning message to indicate the missing data set. Product developers who supply DLLs do not always want or need the IMPORT statements associated with a DLL retained. In these instances, the SYSDEFSD would be better defined as either a temporary or DUMMY data set.

SMP/E is defining a new ddname called SMPDUMMY, which will always be allocated as 'DD DUMMY'. Product packagers may now specify the SYSDEFSD DD statement in the JCLIN input stream as any of the following:

- //SYSDEFSD DD DSN=SMPDUMMY,DISP=xxx
- //SYSDEFSD DD DSN=NULLFILE
- //SYSDEFSD DD DUMMY

In each case, the SIDE DECK LIBRARY subentry of the LMOD entry will be set to SMPDUMMY. When needed for processing, SMPDUMMY will be dynamically allocated by SMP/E as a DUMMY data set.

Coexistence considerations

Unless the required PTF is installed, SMP/E releases prior to SMP/E V3R2 cannot process SYSMOD input that uses the SYSDEFSD DUMMY enhancement, nor can they process the data entries for these elements.

For more information about SYSDEFSD DUMMY, refer to *SMP/E Reference*.

SMP/E Dialog Customization

A new option, Option 0, has been added to the SMP/E Primary Option Menu GIM@PRIM to implement the current SMP/E customization options. This new option allows you to enter or change the values for the customization options that were previously found in panel GIM@UPRM. When you select option 0 from the GIM@PRIM panel, the panel GIM@PARM will appear. The options you then specify are saved permanently in the ISPF profile pool for later use by other SMP/E dialog processes.

Migration tasks

All dialog customization formerly specified on panel GIM@UPRM must now be specified using Option 0 on the SMP/E Primary Option Menu. When you move to a new release of SMP/E and continue to use the same ISPF profile data set, no migration actions are required to use the options previously entered and saved.

For more information about SMP/E dialog customization, refer to the tutorial panels that accompany the SMP/E dialogs.

GIMUTTBL Removal

Module GIMUTTBL and load module GIMUTTBL are no longer supplied as part of SMP/E. Macro GIMDFUT, which was used to replace the IBM-supplied copy of GIMUTTBL, is also no longer supplied. GIMUTTBL was formerly used to specify which utility programs SMP/E can call.

Migration tasks

You can specify which utility programs SMP/E can call by using the PROGRAM class of the z/OS Security Server (RACF). Refer to *z/OS Security Server RACF Security Administrator's Guide*, for more information about how to use this function.

SMP/E V3R1 Overview

The following sections describe the new and changed SMP/E functions that are introduced for SMP/E V3R1. The information about each item includes:

- Description
- Summary of the SMP/E tasks or interfaces that might be affected
- Coexistence considerations, if any, that are associated with the item
- Migration procedures, if any, that are associated with the item
- References to other publications that contain additional detailed information

Defining Exit Routines using SMPPARM Member GIMEXITS

SMP/E V3R1 allows you to define exit routines that are to be given control during SMP/E processing by specifying new GIMEXITS control statements in SMPPARM member GIMEXITS. This replaces the previous method of updating module GIMMPUXD. Putting the exit routine information in an SMPPARM member means that the information is persistent and that you do not need to update module GIMMPUXD every time a new release of SMP/E is installed.

Migration tasks

If you have existing exit routines defined in GIMMPUXD or wish to create new exit routines, you must define them in a GIMEXITS member. Any exit routines defined in GIMMPUXD will be ignored by SMP/E

For more detailed information about this change, refer to *SMP/E Reference*.

Dynamic Allocation using SMPPARM Member GIMDDALC

SMP/E V3R1 allows you to define data sets to be dynamically allocated by SMP/E by specifying new GIMDDALC control statements in SMPPARM member GIMDDALC, as well as by using DDDEF entries. This replaces the previous method of updating module GIMMPDFT. Putting the dynamic allocation information in an SMPPARM member means that the information is persistent and that you do not need to update module GIMMPDFT every time a new release of SMP/E is installed.

Migration tasks

If you have used GIMMPDFT to define data sets for dynamic allocation, you must create new definitions in GIMDDALC. SMP/E will ignore any definitions in GIMMPDFT.

For more information about this change, refer to *SMP/E Reference*.

Enhanced Link Name Values

SMP/E V3R1 has increased the maximum length allowed for a hierarchical file system element LINK value from 64 characters to 1023 characters. SMP/E has also increased the maximum length allowed for the alias values on ALIAS link-edit control statements from 64 characters to 1023 characters.

Coexistence considerations

Unless the required PTF is installed, SMP/E releases prior to SMP/E V3R1 cannot process a SYSMOD containing a hierarchical file system element MCS that specifies a LINK value longer than 64 characters, nor can they process a hierarchical file system element entry containing a LINK subentry, or an LMOD entry containing an LMODALIAS subentry, created by SMP/E V3R1, or higher. If installed, the PTF will update the SMP/E Query dialogs and the GIMAPI to retrieve and display information about LINK or LMODALIAS subentries created by SMP/E V3R1, or higher. For all other SMP/E processing, the PTF will cause SMP/E to issue an error message if it encounters an unsupported LINK value or LINK or LMODALIAS subentry.

Migration tasks

An application program that uses the SMP/E Alias Record Type 0 (A0) library change record may need to be updated to handle alias and link values of up to 1023 characters to avoid truncating data. For more information about this change, refer to the chapter on Library Change File Records in *SMP/E Reference*.

An application program that uses the GIMAPI to query the LINK subentry of an hierarchical file system element entry or the LMODALIAS subentry of an LMOD entry may need to be updated to allow for the longer length of these subentries. For more information about this change, refer to the description of these subentries in the GIMAPI chapter of *SMP/E Reference*.

Removal of Function to Create Backup IEANUC01 Load Modules

The ability to save the target system's nucleus load module (IEANUC01) during APPLY, LINK, and RESTORE command processing has been removed from z/OS V1R2 SMP/E. The NUCID operand of the APPLY command and the NUCID subentry are no longer supported and will be ignored by SMP/E if specified.

Migration tasks

An application program that uses the GIMAPI to query the NUCID subentry of an OPTIONS entry must be updated because this subentry no longer exists. For more information about the GIMAPI, refer to the GIMAPI chapter of *SMP/E Reference*.

Conditional JCLIN Processing

SMP/E V3R1 allows a packager to use special JCL comments in the JCLIN input to cause SMP/E to skip over parts of the JCLIN input based on the installation environment. The parts of the JCLIN input that are skipped are not processed by the JCLIN command and do not contribute to the structure information derived from JCLIN processing.

For more information about this change, refer to the section on "Conditional JCLIN Comment Statements" in *SMP/E Commands*, SA22-7771.

Coexistence considerations

Unless the required PTF is installed, SMP/E releases prior to SMP/E V3R1 cannot process JCLIN input that contains the special JCL comments used to skip over parts of the JCLIN input. If installed, the PTF will cause SMP/E to issue an error message if the unsupported JCL comments are encountered.

Network Delivery of SMP/E Input

SMP/E V3R1 can receive input from a network server, in addition to tape and DASD. This enables the delivery of SMP/E-installable products and service over the internet or an intranet. By installing software directly from a network source, SMP/E enables a more seamless integration of electronic software delivery and installation. This reduces the tasks and time required to install software delivered electronically.

SMP/E also provides the GIMZIP and GIMUNZIP service routines to construct, and then later unwrap, network transportable packages of software. This allows you to create your own packages of SMP/E installable software, and then distribute them within your own enterprise, or to other enterprises. Specifically, the GIMZIP service routine will accept partitioned or sequential data sets as input and will create a network transportable package as output. For more information on the GIMZIP and GIMUNZIP service routines, see *SMP/E Reference*, SA22-7772.

Once a package is made accessible on an FTP server, you can use the SMP/E RECEIVE command to transfer the package through a TCP/IP network directly into an SMP/E environment. The RECEIVE command has been extended with new DELETEPKG, FROMNETWORK, and FROMNTS operands to process these network transportable packages. For more information on the RECEIVE command changes, see *SMP/E Commands*, SA22-7771.

New CLIENT and SERVER data sets and SMPDIR and SMPNTS directories have been created to support this new processing. For more information on CLIENT, SERVER, SMPDIR, and SMPNTS, see *SMP/E Reference*, SA22-7772.

Coexistence considerations

Network delivery of SMP/E input requires a new packaging format for that input and new operands on the RECEIVE command. SMP/E releases prior to z/OS SMP/E cannot process this new packaging format and do not recognize the new RECEIVE command operands.

AMODE=64 and COMPAT=PM4 Link Edit Parameters

SMP/E V3R1 recognizes and saves the AMODE=64 and COMPAT=PM4 link edit parameters.

The AMODE option assigns the addressing mode for all of the entry points into a program module (the main entry point, its true aliases, and all of the alternate entry points). AMODE=64 instructs the binder to create AMODE 31/64 executables with 8-byte adcons.

The COMPAT option identifies the compatibility level of the binder. COMPAT=PM4 allows the user to specify a compatibility level appropriate for AMODE=64 executables.

Selected SMP/E Data Sets May Now Reside in the Hierarchical File System

SMP/E V3R1 allows the following data sets to reside in the hierarchical file system:

- SMPCNTL
- SMPDEBUG
- SMPHOLD
- SMPJCLIN
- SMPLIST
- SMPOUT
- SMPPTFIN
- SMPPUNCH
- SMPRPT

For more information about this change, refer to the description of each of these data sets in *SMP/E Reference*.

HFS Data Set Identification

SMP/E V3R1 has enhanced the SMP/E File Allocation Report and SMP/E library change file records to identify the physical HFS data sets where files in the hierarchical file system reside.

SMPPTS Spill Data Sets

SMP/E RECEIVE processing can use SMPPTS spill data sets, if defined, to store SYSMODs when the primary SMPPTS data set is full. Up to 99 spill data sets, named SMPPTS1 through SMPPTS99, can be defined with DD statements or DDDEFs. By eliminating the tasks involved when recovering from an overflowing SMPPTS data set, the use of SMPPTS spill data sets can reduce the amount of manual intervention and data set management required to install software service.

HOLDDATA Summary Reports

SMP/E V3R1 now provides additional HOLDDATA reports for APPLY and ACCEPT processing. The new reports provide you with ++HOLD information in the context of the APPLY or ACCEPT processing output. This frees you from having to manually collect this information, thus saving you significant research time.

SMP/E Load Modules and Service Routines Moved to SYS1.MIGLIB

The SMP/E load modules, service routines, and other SMP/E components that formerly resided in the SYS1.LINKLIB library have been moved to SYS1.MIGLIB. Putting SMP/E into SYS1.MIGLIB enables a driving system to STEPLIB to SMP/E while ensuring that the STEPLIB does not expose the driving system to other executables that are not at the correct level for the driving system.

GIMXTRX Service Routine

GIMXTRX is intended for use as part of an offering called ShopzSeries. It provides two basic functions:

1. Generate a list of target zone names associated with a given GLOBAL zone SMPCSI data set name.
2. Generate a BITMAP representation of FUNCTION and PTF SYSMODs found in a given list of target zone names associated with a given GLOBAL zone SMPCSI data set name.

OS/390 Version 2 Release 7 SMP/E Overview

The following sections describe the new and changed SMP/E functions that are introduced for OS/390 Version 2 Release 7 (V2R7). (This level of SMP/E was also supplied with OS/390 Version 2, Releases 8, 9, and 10.) The information about each item includes:

- Description
- Summary of the SMP/E tasks or interfaces that might be affected
- Coexistence considerations, if any, that are associated with the item
- Migration procedures, if any, that are associated with the item
- References to other publications that contain additional detailed information

SMP/E Planning and Migration Assistant

OS/390 V2R7 (or later) SMP/E provided a Planning and Migration Assistant to assist users in managing their existing system and planning to migrate to a new OS/390 system.

Data Element Reformatting

SMP/E can install data elements during APPLY, ACCEPT, RESTORE, and GENERATE into the output data sets based on their physical attributes.

Coexistence considerations

Releases of SMP/E prior to OS/390 Version 2 Release 7 cannot process the DEIINST and HFSINST jobs created by the GENERATE command of SMP/E V3R1 or z/OS SMP/E. Also, an HFSINST job created by the GENERATE command from a release of SMP/E prior to OS/390 Release 7 cannot be processed by z/OS SMP/E or by OS/390 V2R7 (or later) SMP/E.

Rerun the GENERATE job using the SMP/E release that will be used to process the resulting JCL.

Description for a SYSMOD

OS/390 V2R7 (or later) SMP/E enabled product developers and packagers to include additional descriptive information in a SYSMOD header MCS (that is, in a ++APAR, ++FUNCTION, ++PTF, or ++USERMOD statement).

Improved Protection for Hierarchical File System Files

Before manipulating files in the hierarchical file system, SMP/E temporarily switches the SMP/E user to superuser authority (UID=0) when manipulating files in the hierarchical file system and restores the user to the previous level of authority when the SMP/E updates are done. This means that SMP/E users do not need to have UID=0 (superuser) authority all the time, which reduces the chance of such users accidentally erasing or damaging files in the hierarchical file system while performing non-SMP/E work.

Coexistence considerations

The SMP/E user must be defined to the security class BPX.SUPERUSER for this process to work properly.

Pre-Built Load Module Support

The ++PROGRAM MCS can be used to add, replace, or delete pre-built load modules and program objects in PDS and PDSE data sets as complete entities in functions and PTFs.

Product Data

The ++PRODUCT and ++FEATURE MCS can be used to add, replace, or delete additional product and feature data.

Sequential Data Set Support

SMP/E can now install a data element into a sequential data set.

Coexistence considerations

Releases of SMP/E prior to OS/390 Version 2 Release 7 cannot process the DEIINST and HFSINST jobs created by the GENERATE command of SMP/E V3R1 or z/OS SMP/E. Also, an HFSINST job created by the GENERATE command from a release of SMP/E prior to OS/390 Version 2 Release 7 cannot be processed by z/OS SMP/E or by OS/390 V2R7 (or later) SMP/E.

Rerun the GENERATE job using the SMP/E release that will be used to process the resulting JCL.

Shell Script Support

OS/390 V2R7 (or later) SMP/E enabled the execution of UNIX shell scripts during SMP/E installation of code into the OS/390 UNIX Services environment. SMP/E provides a generic interface to enable a packager to deliver a shell script that can be executed during SMP/E installation, thus reducing the pre-install and post-install requirements of OS/390 UNIX Services application programs.

Symbolic Link Support

Symbolic links can be specified on hierarchical file system MCS.

OS/390 Version 2 Release 5 SMP/E Overview

The following sections describe the new and changed SMP/E functions that are introduced for OS/390 Version 2 Release 5 (V2R5). (This level of SMP/E was also supplied with OS/390 Version 2 Release 6.) The information about each item includes:

- Description
- Summary of the SMP/E tasks or interfaces that might be affected

- Coexistence considerations, if any, that are associated with the item
- Migration procedures, if any, that are associated with the item
- References to other publications that contain additional detailed information

CBIPO dialogs

The CBIPO installation dialogs formerly included with SMP/E were removed from SMP/E in OS/390 V2R5 SMP/E. Customers wishing to install a CBIPO on an OS/390 system may still do so using bootstrap dialogs provided with the CBIPO.

Client Code Installation

OS/390 V2R5 (or later) SMP/E provides facilities to simplify the installation of cooperative or client/server products (such as OS/2). This is done with a common SMP/E packaging structure, a common S/390 server repository for client components, and a server repository accessible from any client platform. These facilities allow, for example, storing the client parts in a hierarchical file system (HFS) and thereby allowing them to be packaged and installed along with the host parts, rather than separately.

Coexistence considerations

Enhanced hierarchical file system element MCS cannot be used by SMP/E releases prior to OS/390 V2R5 SMP/E, unless the required PTF is installed.

Global Zone Merge

OS/390 V2R5 (or later) SMP/E provides a method to merge information from one global zone into another global zone, which customers can use to reduce the number of global zones that they must manage. The merged information includes:

- SYSMOD and HOLDDATA entries,
- SYSMOD members in the SMPPTS data set,
- OPTIONS, UTILITY, DDDEF, ZONESET, and FMIDSET entries
- Global zone entry information, such as zone indexes, FMID list, and SRELs.

This facility is useful to customers who use ServerPac.

Library Change Interface

OS/390 V2R5 (or later) SMP/E provides a programming interface that can be used to obtain a synopsis of SMP/E APPLY and RESTORE processing at the library or member level. Customers can use this interface to propagate the libraries and members modified by SMP/E APPLY and RESTORE processing to other systems requiring the changes, thereby facilitating the integration of SMP/E-managed information in multisystem environments.

Coexistence considerations

OPTIONS entries that contain the CHANGEFILE subentry cannot be used by SMP/E releases prior to OS/390 V2R5 SMP/E, unless the required PTF is installed.

Improved Load Module Build Processing

OS/390 V2R5 (or later) SMP/E will not build a load module if SMP/E cannot include all of the load module's component modules that have been installed or are being installed. If such a load module can not be completely built, SMP/E terminates APPLY processing for all affected SYSMODs. In addition, OS/390 V2R5 (or later) SMP/E reduces the likelihood of termination owing to incomplete load

modules by expanding its search for the component modules to include copies of modules from within previously installed SYSMODs in the SMPPTS data set.

Load module return code

OS/390 V2R5 (or later) SMP/E allows product packagers to provide information in the JCLIN to identify the highest return code allowable for each load module. IBM will provide toleration PTFs for this function for prior releases of OS/390 and currently supported releases of SMP/E.

Coexistence considerations

LMOD entries that contain the RETURN CODE subentry cannot be used by SMP/E releases prior to OS/390 V2R5 SMP/E, unless the required PTF is installed.

Performance Improvements

OS/390 V2R5 (or later) SMP/E provides for multitasking of link-edit operations during APPLY, ACCEPT, and RESTORE processing.

PTF Compaction in SMPPTS Data Set

OS/390 V2R5 (or later) SMP/E compacts the SYSMOD (PTF) data within the SMPPTS data set to reduce its size. This ability to compact PTF data prior to release by IBM also allows IBM to shrink the size of the OS/390 service stream, thus reducing the amount of physical media (tapes) required to distribute service, as well as shrink the size of service distributed through various electronic mediums.

Coexistence considerations

- Compacted SMPPTS data created by OS/390 V2R5 (or later) SMP/E cannot be used by releases prior to OS/390 V2R5 SMP/E, unless the required PTF is installed.
- OPTIONS entries that contain the COMPACT subentry cannot be used by SMP/E releases prior to OS/390 V2R5 SMP/E, unless the required PTF is installed.

Enhanced RECEIVE Command Processing

OS/390 V2R5 (or later) SMP/E enables users to prevent the RECEIVE command from processing SYSMODs that are already applied or accepted. Users can specify this with the OPTIONS entry, on the RECEIVE command, or both. This enhancement reduces the need for the user to manually manage the SMPPTS with REJECT commands.

Coexistence considerations

OPTIONS entries that contain the RECZGRP and RECEXZGRP subentries cannot be used by SMP/E releases prior to OS/390 V2R5 SMP/E.

Reduced SMP/E Message Output

OS/390 V2R5 (or later) SMP/E reduced the number of messages issued during APPLY, ACCEPT, and RESTORE processing for easier identification of potential problems. Also, users may specify that messages issued to SMPOUT be formatted to an 80 character width, instead of the previous 120 character width, to make the messages easier to view when displayed on a terminal screen.

Coexistence considerations

OPTIONS entries that contain the MSGWIDTH and MSGFILTER subentries cannot be used by SMP/E releases prior to OS/390 V2R5 SMP/E.

GIMAPI: All Entries and Subentries Support

For OS/390 V2R5 (or later) SMP/E, an application program using the GIMAPI QUERY command may specify an asterisk (*) on entry and subentry parameters to retrieve the Consolidated Software Inventory (CSI) data for all entry types, all subentries, or both.

GIMAPI: Version Support

OS/390 V2R5 (or later) SMP/E can supply to an application program the version of GIMAPI being executed to retrieve information from the CSI. This allows the application program to determine whether information stored in the CSI is supported with the level of the QUERY program that is being executed.

Coexistence considerations

Application programs cannot use the VERSION command of the GIMAPI programming interface on releases of SMP/E prior to OS/390 V2R5 (or later) SMP/E, unless the required PTF is installed.

OS/390 Version 1 Release 3 SMP/E Overview

The following sections describe the new and changed SMP/E functions that are introduced for OS/390 Version 1 Release 3 (V1R3). (This level of SMP/E was also supplied with OS/390 Version 2 Release 4.) The information about each item includes:

- Description
- Summary of the SMP/E tasks or interfaces that might be affected
- Coexistence considerations, if any, that are associated with the item
- Migration procedures, if any, that are associated with the item
- References to other publications that contain additional detailed information

API for User Access to the CSI

A programming interface is provided for read only access to SMP/E's consolidated software inventory (CSI) data. The data in CSI can be used to further automate systems management tasks.

A program called GIMAPI is used to invoke the API. The function can be called from different languages. Examples are provided for C/370 and PL/I.

The following commands are used with the GIMAPI call:

QUERY Request data from the SMP/E CSI and return it to the calling program.

FREE Free storage allocated by invocations of the QUERY command.

Enhanced Cross-Zone Requisite Checking

Cross-zone requisite checking is enhanced. Immediate feedback from the **APPLY**, **ACCEPT**, and **RESTORE** commands assists you in verifying that cross-zone requisites are installed and satisfied.

Optional parameters with these commands provide you the flexibility to:

- Override SMP/E's default method for determining which zones are checked for cross-zone requisites
- Install unsatisfied cross-zone requisites into the set-to zone

- Lessen the severity of a missing cross-zone requisite to a warning versus a terminating error

Coexistence considerations

Releases of SMP/E prior to OS/390 V1R3 SMP/E cannot perform the cross-zone requisite checking requested by the XZREQCHK(YES) subentry of a ZONESET entry and will ignore the request.

Enhanced Exception SYSMOD Report

The enhanced Exception SYSMOD Report, available as a small programming enhancement (SPE) for OS/390 V1R3 SMP/E, includes IBM OS/390 Enhanced HOLDDATA that is provided in ++HOLD statements. The report is reformatted so that it is ordered by FMID within each requested zone. (Previously, the report was ordered by SYSMOD within each zone.) A summary section is placed at the end of the report. The enhanced REPORT ERRSYSMODS command continues to work with legacy HOLDDATA.

The REPORT ERRSYSMODS command has also been enhanced by this SPE to handle held SYSMODs. Previously, a held, resolving SYSMOD was placed in the SMPPUNCH output, but was commented out. The customer had to rerun REPORT ERRSYSMODS command against the GLOBAL zone to determine if the held, resolving SYSMOD had an available resolving SYSMOD. REPORT ERRSYSMODS does this research for the customer and produces one SMPPUNCH output.

Coexistence considerations

- The REPORT ERRSYSMODS command in SMP/E releases prior to OS/390 V1R3 cannot display IBM z/OS Enhanced HOLDDATA as intended. OS/390 V1R3 and V2R4 require the installation of the appropriate SPE.
- The format of the HOLDDATA provided by the SMARTMVS service in Europe or the Electronic HOLDDATA service in the U.S. is not compatible with z/OS Enhanced HOLDDATA and does not take advantage of the enhanced REPORT ERRSYSMODS command. Customers who currently use these services, and who wish to make full use of the REPORT ERRSYSMODS command, must refresh their CSI's HOLDDATA with z/OS Enhanced HOLDDATA.

Enhanced ++IF FMID Processing

z/OS SMP/E allows the ++IF MCS FMID operand to specify the same value as the value of the FMID operand of the previous ++VER MCS (if the SYSMOD is not a base function) or the name of the SYSMOD (if the SYSMOD is a base function).

Enhanced Internal HOLD SYS Processing

Analysis of internal HOLD information when one SYSMOD supersedes another is simplified. When a SYSMOD has ++HOLD information and it is superseded by another SYSMOD, the ++HOLD can be brought forward unchanged. The SYSMOD ID on the ++HOLD need not change to that of the superseding SYSMOD. Even if the SYSMOD ID on the ++HOLD is not the same as the containing SYSMOD, the ++HOLD is effective only against the SYSMOD that contains it. If the SYSMOD ID on the ++HOLD is not the same as the containing SYSMOD, SMP/E can determine if internal HOLDs are satisfied during APPLY and ACCEPT processing and thereby eliminate manual analysis.

Coexistence considerations

SYSMODs that contain a ++HOLD MCS that specifies a SYSMOD ID that is superseded on a preceding ++VER MCS cannot be processed by previous releases of SMP/E during RECEIVE processing, unless the appropriate PTF is installed for the prior release.

Enhanced ZONEEDIT Command

The ZONEEDIT command is enhanced to provide a simplified method of changing path names. A PATH subentry is included on the unconditional CHANGE statement of the ZONEEDIT DDDEF command.

An example of when you might want to use the PATH subentry on the CHANGE statement is to modify path names of DDDEFs during the z/OS OpenEdition service process.

Enhancements to the Binder Utility in DFSMS/MVS

SMP/E can use enhancements to the binder utility in DFSMS/MVS. The enhancements to the binder include elimination of the LE/370 prelinker utility, and building dynamic load library (DLL) program objects. SMP/E's support includes:

- New link-edit parameters are recognized on the LEPARM operand of the ++MOD MCS and in JCLIN used to define a load module. The new parameters are ALIASES, DYNAM, FILL, HOBSET, REUS(NONE|REFR|RENT|SERIAL), RMODE=SPLIT, and UPCASE(YES|NO). All of these new parameters can be specified in JCLIN and all except ALIASES and DYNAM can be specified on the LEPARM operand.
- SMP/E supports the binder in dynamically building a definition side deck file for DLL program objects when those program objects are installed. The library to contain the definition side deck file is identified by a new side deck library (SIDEDECKLIB) subentry in the LMOD entry.
- Load modules that use DLLs can reference the definition side deck files associated with the DLLs. This is done by including the definition side deck files during a link-edit operation. The LMOD entry will contain a new utility input (UTIN) subentry list to record definition side deck files to be included during a link-edit operation.

Coexistence considerations

Releases of SMP/E prior to OS/390 V1R3 SMP/E cannot:

- Correctly install products and service that were developed for installation using the INCLUDE statements in JCLIN that identify UTILITY INPUT for a load module, the SYSDEFSD DD statements in JCLIN that identify the definition side deck library for a load module, and the FILL, HOBSET, RMODE=SPLIT, and EXITS link edit attributes on the LEPARM operand on the ++MOD MCS and in JCLIN.
- Use target and distribution zones containing LMOD or MOD entries updated by OS/390 V1R3 (or later) with FILL, HOBSET, RMODE=SPLIT, EXITS, ALIASES, or DYNAM link edit parameters in the MOD and LMOD entries, the UTILITY INPUT subentry list, or the SIDE DECK LIBRARY subentry in the LMOD entries.

System/390 Service Update Facility

The System/390 Service Update Facility (SUF) available as a small programming enhancement (SPE) for OS/390 V1R3 SMP/E, provides, along with other

System/390 products, provides a common tool across multiple platforms to help customers to maintain their systems with System/390 service facilities.

OS/390 Version 1 Release 2 SMP/E Overview

The following sections describe the new and changed SMP/E functions that are introduced for OS/390 Version 1 Release 2 (V1R2). The information about each item includes:

- Description
- Summary of the SMP/E tasks or interfaces that might be affected
- Coexistence considerations, if any, that are associated with the item
- Migration procedures, if any, that are associated with the item
- References to other publications that contain additional detailed information

BLOCKSIZE=8800 for SMP/E Data Sets

For the RELFILE data sets, target libraries, and distribution libraries containing z/OS SMP/E, data sets that are allocated with RECFM=FB and LRECL=80 are allocated with BLKSIZE=8800.

BUILDMCS Command

The BUILDMCS command provides a process for copying products from one pair of target and distribution zones and libraries, to another pair of target and distribution zones and libraries. This command generates the MCS and JCLIN required to reinstall the specified FMIDs.

Coexistence considerations

SYSMOD input (modification control statements) created by the BUILDMCS command cannot be processed by SMP/E releases prior to OS/390 V1R2 SMP/E, unless the required PTF is installed.

Bypassing System Holds for Specific SYSMODs

For APPLY and ACCEPT processing, you can bypass a particular system hold for specific SYSMODs, instead of for all SYSMODs held for that reason ID. For example, a number of SYSMODs might be held because they require you to take some required action before installing them. If you have completed the required action for some (but not all) of the held SYSMODs, you can request SMP/E to bypass that hold reason ID only for the SYSMODs you specify. All other SYSMODs affected by that reason ID remain held.

FMIDSET Selection

SMP/E provides additional granularity of FMIDSET specification on the SELECT operand of the APPLY, ACCEPT, RESTORE, and RECEIVE commands to allow you to install sets of FMIDs.

Receiving Relative File Data Sets Created from PDSEs

When allocating a new SMPTLIB data set during RECEIVE processing, SMP/E checks the format of the associated relative file (RELFILE) data set, then uses the appropriate data set type (LIBRARY or PDS) for the SMPTLIB data set. Here are some benefits of this change:

- When packaging SYSMODs, you can ship program objects in RELFILEs, because SMP/E can load RELFILEs that were created from PDSEs into SMPTLIB data sets that are PDSEs.

- When receiving SYSMODs, you do not have to preallocate SMPTLIB data sets with the appropriate data set type, because SMP/E can allocate the SMPTLIB data set as PDS or LIBRARY, based on the format of the corresponding RELFILE data set.

SMP/E Dialogs: FIND Command

You can use the FIND primary command in the SMP/E dialogs. The FIND command makes it easier for you to quickly locate a specified character string in the table display section of panels in the following dialogs:

- SYSMOD Management
- Query
- Receive

Panels that allow the FIND command state that you can use the command. The help panels for these dialog panels explain how to use the FIND command.

SMP/E GIMOPCDE Member Moved from PARMLIB

The GIMOPCDE member, which SMP/E optionally uses to determine valid OPCODES during the scanning of JCLIN, has been removed from PARMLIB. Instead, a ready-to-use default set of OPCODE definitions is contained within SMP/E.

You may optionally provide an SMPPARM data set, which may contain your own OPCODE member to override the defaults supplied with SMP/E. The user-provided OPCODE member is a text member that you store in a user-allocated PDS named SMPPARM. You are not required to allocate the SMPPARM data set, unless you want to supply your own OPCODE member. If you provide an OPCODE member, it is used instead of SMP/E's default set.

SMP/E also provides a sample text member, named GIMOPCDE, that you can use as a starting point for creating your own OPCODE member.

Summary of Interface Changes

This section summarizes the new and changed interface components of z/OS SMP/E.

Commands

Table 17 lists the changes to the SMP/E commands for this release. See *SMP/E Commands*, SA22-7771 for more detailed information about these commands.

Table 17. Summary of Changed Commands

Command Name	Release	Description	Related Support
APPLY	SMP/E V3R1	**Changed:** NUCID operand removed	"Removal of Function to Create Backup IEANUC01 Load Modules" on page 177
LINK LMODS	SMP/E V3R2	**New:** for load module management	"Smaller SMPLTS data set" on page 174
RECEIVE	SMP/E V3R1	**Changed:** New operands	"Network Delivery of SMP/E Input" on page 178
REPORT CALLLIBS	SMP/E V3R2	**Removed**	"Smaller SMPLTS data set" on page 174

Table 17. Summary of Changed Commands (continued)

Command Name	Release	Description	Related Support
UPGRADE	SMP/E V3R2	**New:** for change control	"UPGRADE Command" on page 172

Data Sets and Files

Table 18 lists the changes to the SMP/E data sets and files for this release. See *SMP/E Reference*, SA22-7772 for more detailed information about these data sets and files.

Table 18. Summary of Changed Data Sets

Data Set	Release	Description	Related Support
SMPWKDIR	SMP/E V3R2	**New:** for GIMZIP, GIMUNZIP, RECEIVE FROMNETWORK, and RECEIVE FROMNTS processing	"GIMZIP: Archive Segmentation" on page 173
SYSDEFSD	SMP/E V3R2	**Changed:** to allocate a DUMMY data set	"DUMMY data set for SYSDEFSD" on page 175
CLIENT	SMP/E V3R1	**New:** for RECEIVE FROMNETWORK processing	"Network Delivery of SMP/E Input" on page 178
SERVER	SMP/E V3R1	**New:** for RECEIVE FROMNETWORK processing	"Network Delivery of SMP/E Input" on page 178
SMPDIR	SMP/E V3R1	**New:** for GIMZIP and GIMUNZIP processing	"Network Delivery of SMP/E Input" on page 178
SMPNTS	SMP/E V3R1	**New:** for RECEIVE FROMNETWORK and RECEIVE FROMNTS processing	"Network Delivery of SMP/E Input" on page 178
SMPPTS	SMP/E V3R1	**Changed:** Spill data sets may be defined	"SMPPTS Spill Data Sets" on page 179
various	SMP/E V3R1	**Changed:** Selected data sets may reside in the hierarchical file system	"Selected SMP/E Data Sets May Now Reside in the Hierarchical File System" on page 179

Exits

Although the methods by which SMP/E exits are invoked has changed (see the entry on GIMEXITS in Table 24 on page 193), no changes have been made to the SMP/E exits themselves. For more information on SMP/E exits, see *SMP/E Reference*.

Macros

Table 19 lists the changes to the SMP/E macros for this release.

Table 19. Summary of Changed Macros

Macro	Release	Description	Related Support
GIMDFUT	SMP/E V3R2	Deleted	"SMP/E Dialog Customization" on page 176

Table 19. Summary of Changed Macros (continued)

Macro	Release	Description	Related Support
GIMMPUXD	SMP/E V3R1	Deleted	"Defining Exit Routines using SMPPARM Member GIMEXITS" on page 176

Messages

This section describes the new and changed SMP/E messages. To determine if updates are required, compare the message identifiers and the corresponding message text with any automated operation procedures your installation uses.

See *SMP/E Messages, Codes, and Diagnosis*, GA22-7770 for detailed information about these SMP/E messages. For information about other z/OS message changes that may affect your installation, refer to *z/OS Summary of Message Changes*.

Panels

Table 20 lists the new and changed SMP/E panels. Some panels may be updated by more than one release enhancement.

Table 20. Summary of New and Changed Panels

Panel Name	Release	Description	Related Support
GIM@PARM	SMP/E V3R2	New	"SMP/E Dialog Customization" on page 176
GIM@PRIM	SMP/E V3R2	Updated	"SMP/E Dialog Customization" on page 176
GIM@UPRM	SMP/E V3R2	Deleted	"SMP/E Dialog Customization" on page 176
GIMQIT28	SMP/E V3R1	Updated	"Enhanced Link Name Values" on page 177
GIMCGRDI	SMP/E V3R1	Updated	"Network Delivery of SMP/E Input" on page 178
GIMCGRVE	SMP/E V3R1	Updated	"Network Delivery of SMP/E Input" on page 178
GIMCGAPA	SMP/E V3R1	Updated	"Removal of Function to Create Backup IEANUC01 Load Modules" on page 177
GIMDFOH	SMP/E V3R1	Updated	"Removal of Function to Create Backup IEANUC01 Load Modules" on page 177
GIMHCAP	SMP/E V3R1	Updated	"Removal of Function to Create Backup IEANUC01 Load Modules" on page 177

Table 20. Summary of New and Changed Panels (continued)

Panel Name	Release	Description	Related Support
GIMHCNU	SMP/E V3R1	Updated	"Removal of Function to Create Backup IEANUC01 Load Modules" on page 177
GIMHIPB0	SMP/E V3R1	Updated	"Removal of Function to Create Backup IEANUC01 Load Modules" on page 177
GIMHIPD1	SMP/E V3R1	Updated	"Removal of Function to Create Backup IEANUC01 Load Modules" on page 177
GIMHIPN	SMP/E V3R1	Updated	"Removal of Function to Create Backup IEANUC01 Load Modules" on page 177
GIMHIP0P	SMP/E V3R1	Updated	"Removal of Function to Create Backup IEANUC01 Load Modules" on page 177
GIMHOH0	SMP/E V3R1	Updated	"Removal of Function to Create Backup IEANUC01 Load Modules" on page 177
GIMHOO0	SMP/E V3R1	Updated	"Removal of Function to Create Backup IEANUC01 Load Modules" on page 177
GIMHQ011	SMP/E V3R1	Updated	"Removal of Function to Create Backup IEANUC01 Load Modules" on page 177
GIMHXA3B	SMP/E V3R1	Updated	"Removal of Function to Create Backup IEANUC01 Load Modules" on page 177
GIMISAPB	SMP/E V3R1	Updated	"Removal of Function to Create Backup IEANUC01 Load Modules" on page 177
GIMISAPD	SMP/E V3R1	Updated	"Removal of Function to Create Backup IEANUC01 Load Modules" on page 177
GIMQIT11	SMP/E V3R1	Updated	"Removal of Function to Create Backup IEANUC01 Load Modules" on page 177
GIMQIX10	SMP/E V3R1	Updated	"Removal of Function to Create Backup IEANUC01 Load Modules" on page 177

Programming Interfaces

Table 21 identifies changes to the SMP/E programming interfaces.

Table 21. Summary of SMP/E Programming Interfaces

Interface	Release	Description	Related Support
GIMMPDFT	SMP/E V3R1	Deleted	"Dynamic Allocation using SMPPARM Member GIMDDALC" on page 177
GIMMPUXD	SMP/E V3R1	Deleted	"Defining Exit Routines using SMPPARM Member GIMEXITS" on page 176
GIMUTTBL	SMP/E V3R2	Deleted	"GIMUTTBL Removal" on page 176
GIMAPI	SMP/E V3R1	**Changed:** length of the LINK and LMODALIAS subentries	"Enhanced Link Name Values" on page 177
GIMAPI	SMP/E V3R1	**Changed:** NUCID subentry deleted	"Removal of Function to Create Backup IEANUC01 Load Modules" on page 177
Library Change File Records	SMP/E V3R1	**Changed:** length of alias and link values in A0 record	"Enhanced Link Name Values" on page 177
Link Edit Parameters	SMP/E V3R1	**Changed:** additional values recognized for AMODE and COMPAT	"AMODE=64 and COMPAT=PM4 Link Edit Parameters" on page 179

Service Routines

Table 22 identifies changes to the SMP/E service routines.

Table 22. Summary of SMP/E Service Routines

Interface	Release	Description	Related Support
GIMUNZIP	SMP/E V3R1	**New:** service routine to unzip network packages	"Network Delivery of SMP/E Input" on page 178
GIMXSID	SMP/E V3R2	**New:** service routine for use with ShopzSeries	"GIMXSID Service Routine" on page 173
GIMXTRX	SMP/E V3R1	**New:** service routine for use with ShopzSeries	"GIMXTRX Service Routine" on page 180
GIMZIP	SMP/E V3R1	**New:** service routine to create network packages	"Network Delivery of SMP/E Input" on page 178
GIMZIP	SMP/E V3R2	**New: subdir** attribute on <FILEDEF> tag to create user-defined subdirectories	"GIMZIP: User Defined Subdirectories" on page 173
GIMZIP	SMP/E V3R2	**New:** ARCHSEG tag to define an archive segment	"GIMZIP: Archive Segmentation" on page 173
GIMZIP	SMP/E V3R2	**New:** SEGMENT option to define the size of archive segments	"GIMZIP: Archive Segmentation" on page 173

SMPPARM Members

Table 23 identifies changes to the SMPPARM members.

Table 23. Summary of SMP/E Changes to SYS1.SAMPLIB

Member Name	Release	Description	Related Support
GIMEXITS	SMP/E V3R1	**New:** Identifies exit routines	"SMP/E Dialog Customization" on page 176

SYS1.SAMPLIB Members

Table 24 identifies changes to the SMP/E members of SYS1.SAMPLIB.

Table 24. Summary of SMP/E Changes to SYS1.SAMPLIB

Member Name	Release	Description	Related Support
GIMASAMP	OS/390 V1R3	Contains sample GIMAPI assembler application program	"API for User Access to the CSI" on page 184
GIMCRSAM	OS/390 V1R3	Contains sample C/370 ″clean-up FMIDs″ program using GIMAPI	
GIMCSAMP	OS/390 V1R3	Contains sample GIMAPI C/390 application program	"API for User Access to the CSI" on page 184
GIMDDALC	SMP/E V3R1	Contains sample statements for data set allocation	"Dynamic Allocation using SMPPARM Member GIMDDALC" on page 177
GIMEXITS	SMP/E V3R1	Contains sample statements for defining exit routines	"Defining Exit Routines using SMPPARM Member GIMEXITS" on page 176
GIMOPCDE	OS/390 V1R2	Contains sample ODCDE statements	"SMP/E GIMOPCDE Member Moved from PARMLIB" on page 188
GIMPRSAM	OS/390 V1R3	Contains sample PL/I ″clean-up FMIDs″ program using GIMAPI	
GIMPSAMP	OS/390 V1R3	Contains sample GIMAPI PL/I application program	"API for User Access to the CSI" on page 184
GIMSAMPU	pre-OS/390	Contains sample UCLIN statements	
GIMSAMPU	SMP/E V3R2	**Updated:** Contains steps to allocate SMPCSI data sets and SMP/E operational data sets (such as SMPPTS and SMPLOG) and UCLIN statements to initialize the newly allocated SMPCSI data sets	

Appendix B. Recommended Service Upgrade (RSU)

Recommended Service Upgrade (RSU) is a preventive service philosophy that applies to z/OS. RSU is intended to reduce the volume of PTFs customers must apply for preventive maintenance and to reduce the chance of encountering a PTF in error (PE), resulting in a more stable system.

IBM recommends that customers APPLY all RSU PTFs as preventive maintenance on their z/OS systems. However, customers must make the final decision as to what maintenance they will install.

The recommended service for the following products is tested in the Consolidated Service Test (CST) cycle:
- z/OS and OS/390
- CICS Transaction Server for OS/390
- DB2 UDB for OS/390
- IMS
- MQSeries for OS/390

Recommended Service Upgrades, with an SMP/E SOURCEID of RSU*yymm*, are available:
- Quarterly, with all PTFs that completed Consolidated Service Test (CST) testing during the prior quarter, including severity 1 through severity 4 APARs.
- Monthly, with high impact or pervasive (HIPER) PTFs, PTF-in-error (PE) PTFs, and other recommended PTFs (recommended because of new function, serviceability, installability, security, or integrity) that have been CST tested.

For information about the latest recommended level of service, see the CST and RSU web site:

`http://www.ibm.com/servers/eserver/zseries/zos/servicetst/`

Note that while all CST-tested PTFs become RSU PTFs, not all RSU PTFs are tested in CST. Only the following are included in CST testing: z/OS, z/OS.e, OS/390, CICS, DB2, IMS, and MQSeries.

RSU is provided for z/OS through ESO, CBPDO, and ServerPac.

Appendix C. Accessibility

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use software products successfully. The major accessibility features in z/OS™ enable users to:

- Use assistive technologies such as screen-readers and screen magnifier software
- Operate specific or equivalent features using only the keyboard
- Customize display attributes such as color, contrast, and font size

Using assistive technologies

Assistive technology products, such as screen-readers, function with the user interfaces found in z/OS. Consult the assistive technology documentation for specific information when using it to access z/OS interfaces.

Keyboard navigation of the user interface

Users can access z/OS user interfaces using TSO/E or ISPF. Refer to *z/OS TSO/E Primer*, *z/OS TSO/E User's Guide*, and *z/OS ISPF User's Guide Volume I* for information about accessing TSO/E and ISPF interfaces. These guides describe how to use TSO/E and ISPF, including the use of keyboard shortcuts or function keys (PF keys). Each guide includes the default settings for the PF keys and explains how to modify their functions.

Notices

This information was developed for products and services offered in the USA.

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
USA

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

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.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Mail Station P300
2455 South Road
Poughkeepsie, NY 12601-5400
USA

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

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

BookManager
CICS
DATABASE 2
DB2
DFSMS
DFSMS/MVS
DFSMSdfp
DFSMSdss
IBM
IBMLink
IMS
OS/2
OS/390
RACF
RETAIN
SP
System/390
SystemPac
VTAM
z/OS

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

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

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

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

Glossary

This glossary defines technical terms and abbreviations used in SMP/E documentation. If you do not find the term you are looking for, refer to the index of the appropriate SMP/E manual or view *IBM Glossary of Computing Terms*, located at:
http://www.ibm.com/networking/nsg/nsgmain.htm

Sequence of Entries: For clarity and consistency of style, this glossary arranges the entries alphabetically on a letter-by-letter basis. In other words, only the letters of the alphabet are used to determine sequence; special characters and spaces between words are ignored.

Organization of Entries: Each entry consists of a single-word or multiple-word term or the abbreviation or acronym for a term, followed by a commentary. A commentary includes one or more items (definitions or references) and is organized as follows:

1. An item number, if the commentary contains two or more items.
2. A usage label, indicating the area of application of the term, for example, "In programming," or "In SMP/E." Absence of a usage label implies that the term is generally applicable to SMP/E, to IBM, or to data processing.
3. A descriptive phrase, stating the basic meaning of the term. The descriptive phrase is assumed to be preceded by "the term is defined as ..." The part of speech being defined is indicated by the opening words of the descriptive phrase: "To ..." indicates a verb, and "Pertaining to ..." indicates a modifier. Any other wording indicates a noun or noun phrase.
4. Annotative sentences, providing additional or explanatory information.
5. References, directing the reader to other entries or items in the dictionary.

References: The following cross-references are used in this glossary:

Contrast with. This refers to a term that has an opposed or substantively different meaning.

Synonym for. This indicates that the term has the same meaning as a preferred term, which is defined in its proper place in the glossary.

Synonymous with. This is a backward reference from a defined term to all other terms that have the same meaning.

See. This refers you to multiple-word terms that have the same last word.

See also. This refers the reader to related terms that have a related, but not synonymous, meaning.

Deprecated term for or **Deprecated abbreviation for**. This indicates that the term or abbreviation should not be used. It refers to a preferred term, which is defined in its proper place in the glossary.

Selection of Terms: A term is a word or group of words to be defined. In this glossary, the singular form of the noun and the infinitive form of the verb are the terms most often selected to be defined. If the term may be abbreviated, the abbreviation is given in parentheses immediately following the term. The abbreviation is also defined in its proper place in the glossary.

A

ACCEPT. The SMP/E command used to install SYSMODs in the distribution libraries.

accept. In SMP/E, to install SYSMODs in the distribution libraries. This is done with the ACCEPT command.

accepted SYSMOD. A SYSMOD that has been successfully installed by the SMP/E ACCEPT command. Accepted SYSMODs do not have the ERROR flag set and are found as SYSMOD entries in the distribution zone.

Access method services (AMS). The system utility program used to support VSAM data sets.

AMS. Access method services.

APAR. Authorized program analysis report.

APAR fix. A temporary correction of a defect in an IBM system control program or licensed program that affects a specific user. An APAR fix is usually replaced later by a permanent correction called a PTF. APAR fixes are identified to SMP/E by the ++APAR statement.

applied SYSMOD. A SYSMOD that has been successfully processed by the SMP/E APPLY command. Applied SYSMODs do not have the ERROR flag set and are found as SYSMOD entries in the target zone.

APPLY. The SMP/E command used to install SYSMODs in the target libraries.

apply. In SMP/E, to install SYSMODs in the target libraries. This is done with the APPLY command.

archive. An archive is a network transportable file containing two files; a file attribute file (FAF) and the data file that the FAF describes.

ASSEM entry. An SMP/E entry containing assembler statements that can be assembled to create an object module.

authorized program analysis report (APAR). A report of a problem caused by a suspected defect in a current unaltered release of a program. The correction is called an APAR fix.

B

BACKUP entries. A collection of SMP/E target zone entries that are copied into the SMPSCDS data set during APPLY processing before they are updated by inline JCLIN, a ++MOVE MCS, or a ++RENAME MCS, or before they are deleted by an element MCS with the DELETE operand.

BACKUP entries consist of:

- A SYSMOD entry indicating what entries have been added, deleted, or updated
- ASSEM entries for updated target zone ASSEM entries
- LMOD entries for updated target zone LMOD entries
- MAC entries for updated or deleted target zone MAC entries
- MOD entries for updated or deleted target zone MOD entries
- SRC entries for updated or deleted target zone SRC entries
- Data element entries for deleted target zone data element entries
- DLIB entries for updated target zone DLIB entries

BALR. Branch and link register commands.

base function. A SYSMOD defining elements of the base system or other products that were not previously present in the target libraries. Base functions are identified to SMP/E by the ++FUNCTION statement. SMP/E is an example of a base function.

base level system. The level of the target system modules, macros, source, and DLIBs created by system generation, to which function and service modifications are applicable.

base version of a load module. Some load modules include modules both explicitly (through INCLUDE statements) and implicitly (through a SYSLIB allocation). The base version of such a load module includes only the explicitly-defined modules for the load module. It is maintained by SMP/E in the SMPLTS data set. The base version of a load module is used with the SYSLIB allocation as input to the link-edit utility in order to build the load module in its target libraries.

binder. A program that processes the output of language translators and compilers into an executable program (load module). It is part of the DFSMS/MVS element of z/OS.

bypass. In SMP/E, to circumvent errors that would otherwise cause SYSMOD processing to fail. This is done by using the BYPASS operand on an SMP/E command.

C

causer SYSMOD. A SYSMOD identified by root cause analysis to be at the base of errors that caused other SYSMODs to fail. See *root cause analysis*.

CBPDO. MVS Custom-Built Product Delivery Offering.

CICS. Customer Information Control System.

CLEANUP. The SMP/E command used to delete entries from the SMPMTS, SMPSTS, and SMPSCDS data sets after a SYSMOD has been accepted into the related distribution zone.

CNTL. See *SMPCNTL*.

coexisting functions. Functions that meet these requirements: (1) they are for the same system or subsystem and have the same SREL value, (2) they do not delete or supersede each other and are not negative prerequisites, and (3) if they are base functions, they are for different products. See also *conditionally coexisting functions* and *unconditionally coexisting functions*.

comment statements. Special control statements that are coded as JCL comments and which are used to convey information to SMP/E during JCLIN processing.

conditional requisites. Requisites defined by an ++IF statement. These are requisites that must be installed if the functions specified on the ++IF statements are installed.

conditionally coexisting functions. Functions that coexist but do not have to be in the same zone.

consolidated software inventory. See *SMPCSI*.

corequisite SYSMODs. SYSMODs each of which can be installed properly only if the other is present. Corequisites are defined by the REQ operand on the ++VER statement.

corrective service. Any SYSMOD used to selectively fix a system problem. Generally, corrective service refers to APAR fixes.

cross-zone. (1) A target zone other than the set-to zone that defines a load module containing modules from set-to zone. Also called a *TIEDTO zone.* The modules were added to the load module through the SMP/E LINK command. The relationship between the cross-zone and the set-to zone is established through the TIEDTO subentry in their zone definition entries. See also *set-to zone* and *TIEDTO relationship*. (2) Pertaining to relationships between zones, especially as a result of conditional requisites (++IF statements) or LINK processing. See also *cross-zone requisite, cross-zone load module,* and *cross-zone module*.

cross-zone load module. A load module containing modules from a different zone as a result of LINK processing.

cross-zone module. A module included in a load module from a different zone as a result of LINK processing.

cross-zone requisite. A conditional requisite that must be installed in one zone because of another SYSMOD that is installed in a different zone. The REPORT command can be used to check information saved from ++IF statements and determine where any cross-zone requisites should be installed.

CSI. Consolidated software inventory data set. See *SMPCSI*.

D

data element. An element that is not a macro, module, or source—for example, a dialog panel or sample code.

DDDEF entry. An SMP/E entry containing the information SMP/E needs in order to dynamically allocate a particular data set.

DEBUG. The SMP/E command used to obtain additional information for problem determination—for example, to trace messages or take dumps.

debug. In SMP/E, to obtain additional information for problem determination—for example, to trace messages or take dumps. This is done with the DEBUG command.

definition side deck. A file in the hierarchical file system, a member of a partitioned data set, or a sequential data set that contains binder IMPORT control statements.

deleted function. In SMP/E, a function that was removed from the system when another function was installed. This is indicated by the DELBY subentry in the SYSMOD entry for the deleted function. See also *explicitly deleted function* and *implicitly deleted function*.

deleting function. A function that removes other functions from the system. This is done by specifying them on the DELETE operand of the ++VER statement.

dependent function. A function that introduces new elements or redefines elements of the base level system or other products. A dependent function cannot exist without a base function. Dependent functions are identified to SMP/E by the ++FUNCTION statement.

dialog. The interactive support provided by SMP/E through ISPF. Instead of entering specific commands and operands, you can use panels to specify the desired processing.

distribution library (DLIB). A library that contains the master copy of all the elements in a system. A distribution library can be used to create or back up a target library.

distribution zone. In SMP/E, a group of records in a CSI data set that describes the SYSMODs and elements in a distribution library.

DLIB. Distribution library.

DLIB entry. An SMP/E entry describing a distribution library that has been totally copied into a target library.

DLIBZONE entry. An SMP/E entry containing information used by SMP/E to process a specific distribution zone and the associated distribution libraries.

DLL. Dynamic link library

E

EC. Engineering change.

element. In SMP/E, part of a product, such as a macro, module, dialog panel, or sample code.

element MCS. An MCS used to replace or update an element.

element selection. The process by which SMP/E chooses the appropriate changes for an element affected by several SYSMODs being installed at the same time.

entry. In SMP/E, a collection of records in a CSI data set. An entry can be created, updated, or deleted by use of UCL statements.

environment. The functions (FMIDs) installed on a particular system or subsystem (SREL).

ERROR indicator. In SMP/E, an indicator in a target or distribution zone SYSMOD entry that shows that SYSMOD processing failed. The ERROR indicator is set before SMP/E updates any libraries and is reset if processing is successful. If processing fails, it remains set to show that an error occurred.

ESO. Expanded service options.

exception SYSMOD. A SYSMOD that is in error or that requires special processing before it can be installed. ++HOLD and ++RELEASE statements identify exception SYSMODs.

EXCP. Execute channel programs.

expanded service options (ESO). A tape that includes preventive service PTFs. Where available, it replaces PUTs as the vehicle for delivering preventive service. An ESO contains PTFs and ++ASSIGN statements assigning source IDs for the PTFs. In the United States, this tape is available from the IBM Support Center and can be ordered either by subscription or as needed.

explicitly deleted function. A function deleted because it was specified on the DELETE operand of a ++VER statement in another SYSMOD.

exported zone. A zone copied into a sequential data set by use of the SMP/E ZONEEXPORT command.

external HOLDDATA. ++HOLD statements contained in SMPHOLD. Contrast with *internal HOLDDATA*.

F

FAF. file attribute file.

FE. Field engineering.

feature. See *dependent function*.

file attribute file. A file attribute file (FAF) is a file containing control statements that describe the attributes of the file contained in an SMP/E network transportable archive. The FAF is contained within the archive information about how the file was created.

File Transfer Protocol. File Transfer Protocol (FTP) is a protocol that defines the interactions necessary between a client and server to facilitate the exchange of binary and textual data files.

firewall. A firewall is an intermediate server that functions to isolate a secure network from an insecure network.

FTP. File Transfer Protocol.

FMID. Function modification identifier.

FMIDSET. A group of FMIDs to be used in processing an SMP/E command—for example, to indicate that SYSMODs applicable to certain functions should be installed.

FMIDSET entry. An SMP/E entry defining an FMIDSET.

function. In SMP/E, a product (such as a system component or licensed program) that can be installed in a user's system if desired. Functions are identified to SMP/E by the ++FUNCTION statement. Each function must have a unique FMID.

function modification identifier (FMID). The SYSMOD ID of a function SYSMOD. It identifies the function that currently owns a given element.

functionally higher SYSMOD. A SYSMOD that uses the function contained in an earlier SYSMOD (called the *functionally lower SYSMOD*) and contains additional functions as well.

functionally lower SYSMOD. A SYSMOD whose function is also contained in a later SYSMOD (called the *functionally higher SYSMOD*).

G

GENASM. A subentry in the MAC entry that lists the ASSEM or SRC entries that must be assembled if the macro is replaced or updated.

GENERATE. The SMP/E command used to create a job stream that builds a set of target libraries from a set of distribution libraries.

generate. In SMP/E, to create a job stream that builds a set of target libraries from a set of distribution libraries. This is done with the GENERATE command.

GIMUNZIP. An SMP/E service routine used to extract files contained in network transportable packages that were built using GIMZIP.

GIMZIP. An SMP/E service routine used to produce network transportable packages.

global zone. A group of records in a CSI data set used to record information about SYSMODs received for a particular system. The global zone also contains information that (1) enables SMP/E to access target and distribution zones in that system, and (2) enables you to tailor aspects of SMP/E processing.

GLOBALZONE entry. An SMP/E entry containing information that SMP/E uses to process the global zone, the associated target and distribution zones, and SMPPTS.

GTF. Generalized trace facility.

H

hashing. An operation that uses a one-way (irreversible) function on data, usually to reduce the length of the data and to provide a verifiable authentication value (checksum) for the hashed data.

header MCS. An ++APAR, ++FUNCTION, ++PTF, or ++USERMOD statement. The header MCS indicates the type of SYSMOD.

HFS. Hierarchical file system.

hierarchical file system element. An element that has a hierarchical file system as its "target library".

hierarchy. In SMP/E, the top-down structure of function and service SYSMODs, in which each SYSMOD is dependent on the one above it.

higher functional level. An element version that contains all the functions of all other relevant versions of that element.

HOLDDATA. In SMP/E, MCSs used to indicate that certain SYSMODs contain errors or require special processing before they can be installed. ++HOLD and ++RELEASE statements are used to define HOLDDATA. SYSMODs affected by HOLDDATA are called *exception SYSMODs*.

HOLDDATA entry. An SMP/E entry containing ++HOLD statements that either were received from SMPHOLD (external HOLDDATA) or were within a SYSMOD that was received (internal HOLDDATA).

I

ICSF. Integrated Cryptographic Service Facility.

IFREQ. A conditional requisite. Conditional requisites are specified on the REQ operand of the ++IF statement.

IMASPZAP. The system utility program used to install superzaps, which are changes for modules, load modules, or CSECTs within modules.

implicitly deleted function. A function deleted because of its dependency on an explicitly deleted function that is specified on the DELETE operand of the ++VER statement.

imported zone. A zone copied from a sequential data set into another zone by use of the SMP/E ZONEIMPORT command.

IMS. Information Management System.

IMSGEN. IMS generation.

indicator. See *subentry indicator*.

in effect. Having control over SMP/E processing. For example, an OPTIONS entry is in effect if (1) it is specified on the SET command or (2) it is defined as the default OPTIONS entry for the set-to zone.

inline data. Information (such as utility control statements or code for an element) that is packaged directly after the associated MCS, rather than in a separate file or data set.

inline JCLIN. The JCL statements associated with a ++JCLIN statement. Inline JCLIN may immediately follow the ++JCLIN statement, or it may be in the RELFILE or TXLIB data set pointed to by the ++JCLIN statement. Inline JCLIN is used to update the target zone when a SYSMOD is applied, or the distribution zone when a SYSMOD is accepted. Contrast with *JCLIN input*.

inner macro. A macro invoked by another macro. In particular, inner macros are those that SMP/E does not detect during JCLIN processing of assembler job steps.

install. In SMP/E, to apply a SYSMOD to the target libraries or to accept a SYSMOD into the distribution libraries.

internal HOLDDATA. ++HOLD statements contained within a SYSMOD. Contrast with *external HOLDDATA*.

I/O. Input or output.

IOGEN. Input/output device generation.

IPL. Initial program load.

ISMD. IBM Software Manufacturing and Delivery (formerly called *PID*).

ISPF. Interactive System Productivity Facility.

ISPF/PDF. Interactive System Productivity Facility/Program Development Facility.

IVP. Installation verification procedure.

J

JAR. The SMP/E entry or MCS that describes a Java ARchive (JAR) file. Also, the abbreviation for Java ARchive (see *Java ARchive(JAR))*.

JARUPD. The SMP/E MCS used to describe an update to a JAR element.

JCL. Job control language.

JCLIN. (1) The SMP/E command used to process data from SMPJCLIN. (2) The ++JCLIN statement, which is associated with JCLIN data that is included in a SYSMOD. (3) SMPJCLIN. See *SMPJCLIN*.

See also *inline JCLIN* and *JCLIN data*.

JCLIN data. The JCL statements associated with the ++JCLIN statement or saved in the SMPJCLIN data set. They are used by SMP/E to update the target zone when the SYSMOD is applied. Optionally, SMP/E can use JCLIN data to update the distribution zone when the SYSMOD is accepted.

JCLIN input. The JCL statements contained in SMPJCLIN and used as input for the JCLIN command. Contrast with *inline JCLIN*.

jar. The Java command used to invoke the Java Archive Tool. The Java Archive Tool is used to perform operations on Java ARchive (JAR) files.

Java ARchive (JAR). An archive file format based on the ZIP file format. Used for aggregating many Java applet component files into one.

job control language (JCL). A problem-oriented language designed to express statements in a job that are used to identify the job or describe its requirements to an operating system.

L

licensed program. A program that performs a function for the user, and usually interacts with and relies upon the system control program or some other IBM-provided control program. Generally, a licensed program is a software package that can be ordered from the program libraries, such as IBM Software Distribution (ISMD). IMS and CICS are examples of licensed programs.

LINK LMODS. The SMP/E command used to relink load modules that use CALLLIBS.

LINK MODULE. The SMP/E command used to link modules in one zone with load modules in another zone.

link library (LKLIB). A data set containing link-edited object modules.

LIST. The SMP/E command used to display entries in SMP/E data sets.

list. In SMP/E, to display entries in SMP/E data sets. This is done with the LIST command.

LKLIB. Link library.

LMOD. In SMP/E, an abbreviation for load module.

LMOD entry. An SMP/E entry containing all the information needed to replace or update a given load module.

load module. A computer program in a form suitable for loading into main storage for execution. It is usually the output of a link-edit utility.

LOG. (1) The SMP/E command used to write user-supplied information to the SMPLOG data set. (2) The SMPLOG data set. See *SMPLOG*.

lower functional level. An element version that is contained in a later element version.

M

MAC. The SMP/E entry or MCS that describes a macro.

macro. An instruction in a source language that is to be replaced by a defined sequence of instructions in the same source language.

MACUPD. The SMP/E MCS used to update a macro.

mass-mode processing. In SMP/E, processing that includes all eligible SYSMODs, regardless of whether they were individually selected.

master CSI. The CSI data set that contains the global zone.

MCS. Modification control statement.

MCS entry. An SMP/E entry containing a copy of a SYSMOD exactly as it was received from SMPPTFIN. MCS entries are in SMPPTS, which is used to store SYSMODs.

MOD. The SMP/E entry or MCS that describes an object module or a single-module load module.

MODID. Modification identifier.

modification. In SMP/E, an alteration or correction to a system control program, licensed program, or user program. Synonymous with *system modification* (SYSMOD).

modification control statement (MCS). An SMP/E control statement used to package a SYSMOD. MCSs describe the elements of a program and the relationships that program has with other programs that may be installed on the same system.

modification identifier (MODID). A list of SYSMOD IDs, including the last SYSMOD that totally replaced the element (RMID), any subsequent partial updates to the element (UMIDs), and the function that owns the element (FMID). MODIDs are contained in element entries.

modification level. A distribution of all temporary fixes that have been issued since the previous modification level. A change in modification level does not add new functions or change the programming

support category of the release to which it applies. Contrast with *release* and *version*.

Note: Whenever a new release of a program is shipped, the modification level is set to 0. When the release is reshipped with the accumulated services changes incorporated, the modification level is incremented by 1.

module. Synonym for *object module* or *single-module load module*.

MTS. Macro temporary storage data set. See *SMPMTS*.

MTSMAC entry. An SMP/E entry that is a copy of a macro that resides only in a distribution library but is needed temporarily during APPLY processing. MTSMAC entries are in the SMPMTS data set.

MVS. Multiple Virtual Storage.

MVS Custom-Built Product Delivery Offering (CBPDO). A software delivery offering used to add products or service to an existing MVS, NCP, CICS, or IMS system.

N

NCP. Network Control Program.

negative prerequisite (NPRE). In SMP/E, a function that is mutually exclusive with another function. It is defined by the NPRE operand on the ++VER statement.

NPRE. Negative prerequisite.

O

object deck. Object module input to the link-edit utility that is placed in the input stream, in card format.

object module. A module that is the output from a language translator (such as a compiler or an assembler). An object module is in relocatable format with machine code that is not executable. Before an object module can be executed, it must be processed by the link-edit utility.

When an object module is link-edited, a load module is created. Several modules can be link-edited together to create one load module (for example, as part of SMP/E APPLY processing), or an object module can be link-edited by itself to create a single-module load module (for example, to prepare the module for shipment in RELFILE format or in an LKLIB data set or as part of SMP/E ACCEPT processing).

operating system. In SMP/E, the system updated by APPLY and RESTORE processing. It consists of the target libraries. Also called the target system.

OPTIONS entry. An SMP/E entry defining processing options that are to be used by SMP/E.

P

package attribute file. A package attribute file (PAF) is a file that contains control statements describing the contents of a network transportable package.

packaging. Adding the appropriate MCS statements to elements to create a SYSMOD, then putting the SYSMOD in the proper format on the distribution medium, such as a tape or direct access data sets.

PAF. package attribute file.

partitioned data set extended (PDSE). A system-managed data set containing an indexed directory and members that are similar to the directory and members of partitioned data sets. A PDSE can be used instead of a partitioned data set.

PE. See *program error PTF*.

PE-PTF. See *program error PTF*.

PID. The former name for ISD.

PRE. Prerequisite.

prerequisite (PRE). In SMP/E, a SYSMOD that must be installed before or along with another SYSMOD in order for that other SYSMOD to be successfully installed. It is defined by the PRE operand on the ++VER statement.

preventive service. (1) The mass installation of PTFs to avoid rediscoveries of the APARs fixed by those PTFs. (2) The SYSMODs delivered on the program update tape.

preventive service planning (PSP). Installation recommendations and HOLDDATA for a product or a service level. PSP information can be obtained through the IBM Support Center.

product. Generally, a software package, such as a licensed program or a user application. A product can contain one or more functions and can consist of one or more versions and releases.

product version. All the releases for a given version of a product.

program error PTF (PE-PTF). A PTF that has been found to contain an error. A PE-PTF is identified on a ++HOLD ERROR statement, along with the APAR that first reported the error.

program object. An executable program stored in a PDSE program library. It is similar to a load module, but has fewer restrictions. For SMP/E purposes, program objects are referred to as load modules.

program packaging. See *packaging*.

program product. The former term for licensed program.

program temporary fix (PTF). A temporary solution or bypass of a problem that may affect all users and that was diagnosed as the result of a defect in a current unaltered release of the program. In the absence of a new release of a system or component that incorporates the correction, the fix is not temporary but is the permanent and official correction mechanism. New elements can also be defined in a PTF. PTFs are identified to SMP/E by the ++PTF statement.

program update tape (PUT). The former vehicle for preventive service. See *expanded service options*.

PSP. Preventive service planning.

PSW. Program status word.

PTF. Program temporary fix.

PTS. PTF temporary store data set. See *SMPPTS*.

PTFIN. PTF input. See *SMPPTFIN*.

PUT. See *expanded service options*.

R

RACF. Resource Access Control Facility.

RECEIVE. The SMP/E command used to read in SYSMODs and other data from SMPPTFIN and SMPHOLD.

receive. In SMP/E, to read SYSMODs and other data from SMPPTFIN and SMPHOLD and store them on the global zone for subsequent SMP/E processing. This is done with the RECEIVE command.

regressed SYSMOD. A SYSMOD one or more of whose elements are modified by subsequent SYSMODs that are not related to it.

regressing SYSMOD. A SYSMOD that causes regression of previous modifications when it is installed.

regression. In SMP/E, the condition that occurs when an element is changed by a SYSMOD that is not related to SYSMODs that previously modified the element.

REJECT. The SMP/E command used to remove SYSMODs from the global zone and SMPPTS.

reject. In SMP/E, to remove SYSMODs from the global zone and SMPPTS and delete any related SMPTLIB data sets. This is done with the REJECT command.

related installation materials (RIMs). In IBM custom-built offerings, task-oriented documentation, jobs, sample exit routines, procedures, parameters, and examples developed by IBM.

related SYSMOD. A SYSMOD associated with other SYSMODs by the FMID, PRE, REQ, or SUP operands.

related zone. The zone named in the RELATED subentry of a TARGETZONE or DLIBZONE entry. For a target zone, the related zone is generally the distribution zone for the libraries used to create the target libraries. For a distribution zone, the related zone is generally the target zone for the libraries built from the distribution libraries.

relative file (RELFILE) format. A SYSMOD packaging method where elements and JCLIN data are in separate relative files from the MCSs. When SYSMODs are packaged in relative file format there is a file of MCSs for one or more SYSMODs, and one or more relative files containing unloaded source-code data sets and unloaded link-edited data sets containing executable modules. The relative files can be either unloaded files in IEBCOPY format, or they can be partitioned data sets. Relative file format is the typical method used for packaging function SYSMODs.

relative files (RELFILEs). Unloaded files containing modification text and JCL input data associated with a SYSMOD. These files are used to package a SYSMOD in relative file format.

release. A distribution of a new product or new function and APAR fixes for an existing product. Contrast with *modification level* and *version*.

replacement modification identifier (RMID). The SYSMOD ID of the last SYSMOD that completely replaced a given element.

REPORT. The SMP/E command used to obtain information about SYSMODs that have been installed. These are the types of REPORT commands:

- REPORT CROSSZONE: Lists conditional requisites that must be installed in certain zones because of SYSMODs installed in other zones.
- REPORT ERRSYSMODS: Determines whether any SYSMODs already installed are now exception SYSMODs.
- REPORT SOURCEID: Lists the source IDs associated with SYSMODs in the specified zones.
- REPORT SYSMODS: Compares the SYSMODs installed in two target or distribution zones.

requisite. A SYSMOD that must be installed before or at the same time as the SYSMOD being processed. There are several types of requisites:

- Prerequisites, which are specified by the PRE operand on the SYSMOD's ++VER statement

- Corequisites, which are specified by the REQ operand on the SYSMOD's ++VER statement
- Conditional requisites, which are specified by the REQ operand on the SYSMOD's associated ++IF statement

requisite chain. A sequence of SYSMODs that are directly or indirectly identified as requisites for a given SYSMOD, (A SYSMOD may identify other SYSMODs as requisites, which in turn may have requisites of their own. The requisite chain extends from the initial SYSMOD, through the hierarchy of requisites, until no more SYSMODs are found that have requisites.) See *requisite*.

requisite set. The set of all SYSMODs on the requisite chain for a particular SYS

RESETRC. The SMP/E command used to set the return codes for the previous commands to zero, so that SMP/E can process the current command.

RESTORE. The SMP/E command used to remove applied SYSMODs from the target libraries.

restore. In SMP/E, to remove applied SYSMODs from the target libraries by use of the RESTORE command.

restore group. All the SYSMODs that have a direct or indirect relationship with a SYSMOD being restored by use of the GROUP operand.

RIM. Related installation material.

RMID. Replacement modification identifier.

RMF. Resource measurement facility.

root cause analysis. Processing done by SMP/E for the ACCEPT, APPLY, and RESTORE commands to identify causer SYSMODs (SYSMODs whose failure has led to the failure of other SYSMODs). The types of errors SMP/E analyzes to determine causer SYSMODs include the following:

- Held SYSMODs
- Missing requisite SYSMODs
- Utility program failures: copy, update, assembler, link, zap
- Out-of-space conditions: *x*37 abends
- Missing DD statements and other allocation errors
- ID errors (a SYSMOD does not supersede or specify as a prerequisite an RMID or a UMID)
- JCLIN failures (syntax errors)

RPL. Request parameter list.

RTM2WA. Recovery termination manager 2 work area.

S

SCDS. Save control data set. See *SMPSCDS*.

SCP. System control program.

select-mode processing. In SMP/E, processing that includes individually selected SYSMODs.

service. PTFs and APAR fixes.

service level. The FMID, RMID, and UMID values for an element. The service level identifies the owner of the element, the last SYSMOD to replace the element, and all the SYSMODs that have updated the element since it was last replaced.

service order relationship. A relationship among service SYSMODs that is determined by the PRE and SUP operands, and the type of SYSMOD.

service SYSMOD. Any SYSMOD identified by an ++APAR or ++PTF statement.

service update. The integration of available service into the current release of a function. Since this is not a new release of the function, it does not change the function's FMID.

SET. The SMP/E command used to indicate the zone to be processed.

set. In SMP/E, to indicate which zone should be processed by the subsequent commands. This is done with the SET command.

set-to zone. The zone that was specified on the previous SET command and that is currently being processed. Contrast with *cross-zone*.

SHA-1. Secure Hash Algorithm 1.

side deck. See *definition side deck*.

single-module load module. A load module created by link-editing a single object module by itself—for example, to prepare the module for shipment in RELFILE format or in an LKLIB data set or as part of SMP/E ACCEPT processing.

SMPCNTL. The SMP/E data set or file in the hierarchical file system that contains the SMP/E commands to be processed.

SMPCSI. The SMP/E data set that contains information about the structure of a user's system as well as information needed to install the operating system on a user's system. The SMPCSI DD statement refers specifically to the CSI that contains the global zone. This is also called the *master CSI*.

SMPDEBUG. The SMP/E data set or file in the hierarchical file system that contains a dump requested by the DEBUG command. Depending on the operands

specified, it may contain (1) a dump of SMP/E control blocks and storage areas associated with the specified dump points or (2) a dump of the VSAM RPL control block for the specified SMP/E function.

SMPDUMMY. The SMP/E data set used to define a load module's definition side deck library as a DUMMY data set. SMPDUMMY is always allocated by SMP/E as a DUMMY data set.

SMP/E. A program product, or an element of OS/390 or z/OS, used to install software and software changes on z/OS systems. SMP/E consolidates installation data, allows more flexibility in selecting changes to be installed, provides a dialog interface, and supports dynamic allocation of data sets.

SMP/E commands. Commands defining the processing to be done by SMP/E, such as RECEIVE.

SMP/E entry. An entry in an SMP/E data set—for example, a MOD entry in a CSI data set.

SMPHOLD. SMPHOLD is the source for HOLDDATA (++HOLD and ++RELEASE statements) to be processed by the RECEIVE command. SMPHOLD may be a tape file, a data set, or one or more files in the hierarchical file system.

SMPJCLIN. The SMP/E data set or file in the hierarchical file system that contains a job stream of assembly, link-edit, and copy job steps. This data is typically the stage 1 output from the most recent full or partial system generation. However, it may also be other data in a similar format, such as the output of the GENERATE command. This job stream is used as input to the JCLIN command to update or create entries in a target zone.

SMPLIST. The SMP/E data set or file in the hierarchical file system that contains the output of all LIST commands.

SMPLOG. The SMP/E data set that contains time-stamped records of SMP/E processing. The records in this data set can be written automatically by SMP/E or added by the user through the LOG command.

SMPLOGA. A secondary log data set for SMP/E processing. If SMPLOGA is defined, it is automatically used when the SMPLOG data set is full.

SMPLTS. The SMP/E data set used as a target load module library to maintain the base version of a load module that specifies a SYSLIB allocation in order to implicitly include modules.

SMPMTS. The SMP/E data set used as a target library for macros that exist only in a distribution library, such as macros in SYS1.AMODGEN. The SMPMTS enables the current version of these macros to be used for assemblies during APPLY processing.

SMPNTS. The SMPNTS is a directory structure and associated files contained in the hierarchical file system used for temporary storage of network transported packages that were received during SMP/E RECEIVE processing.

SMPOBJ. The SMP/E data set used for source-maintained products. SMPOBJ contains preassembled modules that can be used to avoid reassembling those modules. These modules must be in load module format—that is, in the same format as modules residing in the distribution library.

SMPOUT. The SMP/E data set or file in the hierarchical file system that contains messages issued during SMP/E processing. It may also contain a dump of the VSAM RPL, if a dump was taken. In addition, it may contain LIST output and reports if SMPLIST and SMPRPT are not defined.

SMPPARM. The data set that contains members to define parameters, such as macros, assembler operation codes, GIMDDALC control statements, and exit routines.

SMPPTFIN. SMPPTFIN is the source of SYSMODs and ++ASSIGN statements to be processed by the RECEIVE command. SMPPTFIN may be a tape file, a data set, or one or more files in the hierarchical file system.

SMPPTS. The SMP/E data set that contains SYSMODs received from SMPPTFIN. SMPPTS is the source of SYSMODs that are installed in the target and distribution libraries.

SMPPTS spill data sets. Optional SMP/E data sets that can be used to store SYSMODs when the SMPPTS data set becomes full.

SMPPUNCH. The SMP/E data set or file in the hierarchical file system that contains output from various SMP/E commands. This output generally consists of commands or control statements.

- GENERATE: A job stream for building target libraries
- REPORT: Commands for installing or listing SYSMODs
- UNLOAD: UCLIN statements for recreating the entries that were unloaded

SMPRPT. The SMP/E data set or file in the hierarchical file system that contains the reports produced during SMP/E processing.

SMPSCDS. The SMP/E data set that contains backup copies of target zone entries that are created during APPLY processing. These backup copies are made before the entries are (1) changed by inline JCLIN, a ++MOVE MCS, or a ++RENAME MCS, or (2) deleted by an element MCS with the DELETE operand. The

backup copies are used during RESTORE processing to return the entries to the way they were before APPLY processing.

SMPSNAP. The SMP/E data set that is used for snap dump output. When a severe error such as an abend or severe VSAM return code occurs, SMP/E requests a snap dump of its storage before doing any error recovery. In addition, the DEBUG command can request a snap dump of SMP/E storage when specified messages are issued, or can request a snap dump of control blocks and storage areas associated with a specified dump point.

SMPSTS. The SMP/E data set used as a target library for source that exists only in a distribution library. The SMPSTS enables the current version of this source to be used for assemblies during APPLY processing.

SMPTLIB. The SMP/E data sets used as temporary storage for relative files loaded from SMPPTFIN during RECEIVE processing. The SMPTLIB data sets are deleted when the associated SYSMOD is deleted by REJECT, RESTORE, or ACCEPT processing.

SMPWKDIR. An optional directory in the hierarchical file system used for temporary work files.

SMPWRK1. The SMP/E data set used as temporary storage for macro updates and replacements that will be processed by an update or copy utility program. SMP/E places the input in SMPWRK1 during APPLY and ACCEPT processing before calling the utility.

SMPWRK2. The SMP/E data set used as temporary storage for source updates and source replacements that will be processed by an update or copy utility program. SMP/E places the input in SMPWRK2 during APPLY and ACCEPT processing before calling the utility.

SMPWRK3. The SMP/E data set used as temporary storage for object modules supplied by a SYSMOD, object modules created by assemblies, and zap utility input following ++ZAP statements.

SMPWRK4. The SMP/E data set used as temporary storage for zap utility or link-edit utility input that contains EXPAND control statements.

SMPWRK6. The SMP/E data set used during ACCEPT and APPLY processing as temporary storage for inline replacements for data elements. SMP/E places the input in this data set so that it can be directly accessed and installed by the copy utility or SMP/E.

source. The source statements that constitute the input to a language translator for a particular translation.

source ID. A 1- to 8-character identifier that indicates how a SYSMOD was obtained—for example, from a particular service level in an ESO. A source ID is associated with a specific SYSMOD by the RECEIVE command or the ++ASSIGN statement.

SOURCEID. The operand used to refer to a source ID.

source module. The source statements that constitute the input to a language translator, such as a compiler or an assembler, for a particular translation.

SRC. The SMP/E entry or MCS statement that describes a source.

SRCUPD. The MCS used to update a source.

SREL. System release identifier.

Storage Management Subsystem (SMS). A DFSMS/MVS facility used to automate and centralize the management of storage. Using SMS, a storage administrator describes data allocation characteristics, performance and availability goals, backup and retention requirements, and storage requirements to the system through data class, storage class, management class, storage group, and ACS routine definitions.

STS. Source temporary store data set. See *SMPSTS*.

STSSRC entry. An SMP/E entry that is a copy of source that resides only in a distribution library but is needed temporarily during APPLY processing. STSSRC entries are in the SMPSTS data set.

stub entry. An element entry or LMOD entry that does not contain the basic information SMP/E requires in order to process the element or load module (such as FMID, RMID, or library names), but does contain other information, such as subentries describing cross-zone relationships.

stub load module. A load module that does not contain the modules needed to perform its basic functions, but does contain other modules, such as cross-zone modules.

subentry. A field in an SMP/E entry. Each subentry has associated with it a type and a value. The same subentry type may occur several times in a single entry, each time with a different value. For example, the modules supplied by a PTF are saved as MOD subentries in the PTF's SYSMOD entry. Some subentries occur only once within an entry, such as the FMID subentry in a target zone MOD entry.

subentry indicator. A subentry that does not have a data value associated with it. An example of an indicator is the ERROR indicator in the SYSMOD entry. An indicator is either on or off.

subentry list. Multiple occurrences of the same subentry type in an entry, each with a different value. For example, the modules supplied by a PTF are saved as names in the MOD subentry list within the SYSMOD entry for that PTF.

SUP. Supersede.

superseded-only SYSMOD. A SYSMOD that has not been installed, but that has been superseded by another SYSMOD that has been installed.

superseded SYSMOD. In SMP/E, a SYSMOD that is contained in or replaced by the SYSMOD or requisite set of SYSMODs currently being processed. This is indicated by the SUPBY subentry in the SYSMOD entry for the superseded SYSMOD. A superseded SYSMOD is functionally lower than the SYSMOD that superseded it.

superseding SYSMOD. In SMP/E, a SYSMOD that contains all the functions in another SYSMOD and is recognized as the equivalent of that other SYSMOD. The superseding SYSMOD uses SUP operand on its ++VER statement to specify the superseded SYSMOD.

superzap. A generic term for the process performed by IMASPZAP. It can also refer to the module updates processed by IMASPZAP.

SVC. Supervised call.

SVRB. Supervisor request block.

SYSGEN. System generation.

SYSLIB. (1) A subentry used to identify the target library in which an element is installed. (2) A concatenation of macro libraries to be used by the assembler. (3) A set of routines used by the link-edit utility to resolve unresolved external references.

SYSMOD. System modification.

SYSMOD entry. An SMP/E entry containing information about a SYSMOD that has been received into SMPPTS, accepted into the distribution libraries, or applied to the target libraries.

SYSMOD ID. System modification identifier.

SYSMOD packaging. See *packaging*.

SYSMOD selection. The process of determining which SYSMODs are eligible to be processed.

SYSPRINT. The data set that contains output from the utilities called by SMP/E.

SYSPUNCH. The temporary data set containing object modules assembled by running the job stream produced by system generation or the GENERATE command. These modules are not installed in the distribution libraries at ACCEPT time.

system control program (SCP). IBM-supplied programming that is fundamental to the operation and maintenance of the system. It serves as an interface with licensed programs and user programs and is available without additional charge.

system generation (SYSGEN). The process of selecting optional parts of an operating system and of creating a particular operating system tailored to the requirements of a data processing installation.

system modification (SYSMOD). The input data to SMP/E that defines the introduction, replacement, or update of elements in the operating system and associated distribution libraries to be installed under the control of SMP/E. A system modification is defined by a set of MCS.

system modification identifier (SYSMOD ID). The name that SMP/E associates with a system modification. It is specified on the ++APAR, ++FUNCTION, ++PTF, or ++USERMOD statement.

system release identifier (SREL). A 4-byte value representing the system or subsystem, such as Z038 for MVS-based products.

SYSUT1, SYSUT2, SYSUT3. Scratch data sets for SMP/E and the utilities it calls.

SYSUT4. A data set that is used instead of the SYSIN data sets when certain utilities are called.

T

target library. A library containing the executable code that makes up a system.

target system. The system updated during APPLY and RESTORE processing, also referred to as the operating system. See also target libraries.

target zone. In SMP/E, a group of records in a CSI data set that describes the SYSMODs, elements, and load modules in a target library.

TARGETZONE entry. An SMP/E entry containing information used by SMP/E to process a specific target zone and the associated target libraries.

TCP/IP. Transmission Control Protocol/Internet Protocol (TCP/IP) is a hardware independent communication protocol used between physically separated computers. It was designed to facilitate communication between computers located on different physical networks.

temporary data set. A work data set (SMPWRK1–SMPWRK6) or utility data set (SYSUT1–SYSUT4). Temporary data sets are allocated when processing for an SMP/E command begins, and deleted when processing is finished.

text library (TXLIB). A data set containing JCLIN input or replacements for macros, source, or object modules that have not been link-edited. It is used when the JCLIN or elements are provided in partitioned data sets rather than inline or in relative files.

TGTLIB. Target library.

TIEDTO relationship. A cross-zone relationship between two target zones created when the LINK command updates a load module in one of the zones to include modules from the other zone. This relationship is established through the TIEDTO subentry in the zone definition entries for each of the zones.

TIEDTO zone. See *cross-zone*.

TLIB. Temporary library. See *SMPTLIB*.

transformed data. Data processed by the GIMDTS service routine so that it can be packaged inline in fixed-block 80 records.

TSO. Time-sharing option.

TXLIB. Text library.

U

UCL. Update control language.

UCL statement. An SMP/E control statement used to define or change information in an SMP/E data set entry. UCL statements are coded between the UCLIN and ENDUCL commands. The UCL statement specifies the action to be taken (ADD, REP, or DEL), the entry to be modified, and any indicators and subentries to be changed.

UCLIN. The SMP/E command used to mark the beginning of UCL statements, which are used to make changes to entries in SMP/E data sets.

UMID. Update modification identifier.

unconditionally coexisting functions. Functions that coexist and must be in the same zone.

UNLOAD. The SMP/E command used to copy data out of SMP/E data set entries in the form of UCL statements.

unload. In SMP/E, to copy data out of SMP/E data set entries in the form of UCL statements, by use of the UNLOAD command.

update. In SMP/E, to change an existing element without replacing it.

update modification identifier (UMID). The SYSMOD ID of a SYSMOD that updated the last replacement of a given element.

user modification (USERMOD). A change constructed by a user to modify an existing function, add to an existing function, or add a user-defined function. USERMODs are identified to SMP/E by the ++USERMOD statement.

USERMOD. User modification.

UTILITY entry. An SMP/E entry containing information used by SMP/E to invoke a particular system utility program.

V

VERSION. An operand on the ++VER or element statement. VERSION specifies one or more SYSMODs containing elements that are functionally lower than elements in the SYSMOD that specifies the operand. The VERSION operand is also used to change ownership of elements.

version. A separate licensed program that is based on an existing licensed program and that usually has significant new code or new functions. Contrast with *release* and *modification level*.

versioned element. An element that is part of more than one function—for example, one that is part of a base function and a dependent function.

VSAM. Virtual Storage Access Method.

VTOC. Volume table of contents.

Z

ZAP. (1) The SMP/E MCS used to package an update for an object module. (2) The superzap control statement used to update an object module. (3) A shortened name for the superzap utility, which is used to install these updates. See *IMASPZAP*.

zone. A partition in a CSI data set.

ZONECOPY. The SMP/E command used to copy a zone from one CSI data set to another.

ZONEDELETE. The SMP/E command used to delete a zone from a CSI data set.

ZONEEDIT. The SMP/E command used to change the values for a subentry in all the DDDEF or UTILITY entries in a given zone.

ZONEEXPORT. The SMP/E command used to copy a zone into a sequential data set.

ZONEIMPORT. The SMP/E command used to load an exported zone from a sequential data set into another zone.

ZONEMERGE. The SMP/E command used to copy one zone into another, or to merge two zones into one.

ZONERENAME. The SMP/E command used to change the name of a zone.

ZONESET. A group of zones to be used when processing an SMP/E command. For example, it may

define the zones that the REPORT command is to check for cross-zone requisites. A ZONESET may also define a group of zones to be checked or ignored by the REJECT command.

ZONESET entry. An SMP/E entry defining a ZONESET.

z/OS UNIX System Services (z/OS UNIX). The set of functions provided by the Shell and Utilities, kernel, debugger, file system, C/C++ Run-Time Library, Language Environment, and other elements of the z/OS operating system that allow users to write and run application programs that conform to UNIX standards.

Index

Special characters

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Z

Readers' Comments — We'd Like to Hear from You

IBM SMP/E for z/OS and OS/390
User's Guide

Publication No. SA22-7773-04

Overall, how satisfied are you with the information in this book?

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Overall satisfaction	☐	☐	☐	☐	☐

How satisfied are you that the information in this book is:

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Accurate	☐	☐	☐	☐	☐
Complete	☐	☐	☐	☐	☐
Easy to find	☐	☐	☐	☐	☐
Easy to understand	☐	☐	☐	☐	☐
Well organized	☐	☐	☐	☐	☐
Applicable to your tasks	☐	☐	☐	☐	☐

Please tell us how we can improve this book:

Thank you for your responses. May we contact you? ☐ Yes ☐ No

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

Name _____ Address _____

Company or Organization _____

Phone No. _____

Readers' Comments — We'd Like to Hear from You
SA22-7773-04

IBM

Cut or Fold
Along Line

Fold and Tape | **Please do not staple** | Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

IBM Corporation
Department 55JA, Mail Station P384
2455 South Road
Poughkeepsie, NY
12601-5400

Fold and Tape | **Please do not staple** | Fold and Tape

SA22-7773-04

Cut or Fold
Along Line

Program Number: 5655-G44

Printed in U.S.A.

SA22-7773-04