

VSE/System Package Migration Volume 2

Version 2 Release 1

**Program Number 5666-316
Order Number SC33-6179-1
File No. S370/4300-36**

Second Edition (March 1985)

This edition is a major revision of SC33-6179-0, an edition which IBM has not made generally available. This edition applies to IBM Virtual Storage Extended/ System Package Version 2 Release 1 (IBM VSE/SP 2.1), Program Number 5666-316, and to all subsequent releases and modifications until otherwise indicated in new editions. Changes are made periodically to the information herein; before using this publication in connection with the operation of IBM systems, consult the latest IBM System/370, 30XX and 4300 Processors Bibliography, GC20-0001, for the editions that are applicable and current.

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM program product in this document is not intended to state or imply that only IBM's program product may be used. Any functionally equivalent program may be used instead.

Publications are not stocked at the addresses given below; requests for copies of IBM publications should be made to your IBM representative or to the IBM branch office serving your locality.

A form for readers' comments is provided at the back of this publication. If the form has been removed, comments may be addressed either to:

IBM Corporation
Dept. 6R1
180 Kost Road
Mechanicsburg, PA 17055, USA

or to:

IBM Deutschland GmbH
Dept. 3282
Schoenaicher Strasse 220
D-7030 Boeblingen, Federal Republic of Germany

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

PREFACE

Both volumes of this publication are intended for the more experienced user who is responsible for planning and carrying out the migration task from a previous release of VSE to VSE/SP 2.1.

Volume 1 describes the tasks necessary before and after migration. The Table of Contents of that manual is listed in the appendix for your convenience.

CONTENTS

Chapter 1. Introduction	1
How to Use This Manual	1
Chapter 2. Interactive Interface	3
Overview	3
System Administration	3
Path Summary IESEADM	3
Keyword Summary IESEADM	6
Application Development	12
Path Summary IESEPROG	12
Keyword Summary IESEPROG	13
System Operation	15
Path Summary IESEOPER	15
Keyword Summary IESEOPER	16
Migrate VSE/ICCF User Profiles	17
Chapter 3. System Organization	21
Highlights	21
ICCF Generation	21
POWER Generation	23
Tailor Libraries / Files	24
Highlights	24
Libraries	27
In BAM Space	27
In VSAM Managed Space	28
ICCF DTSFILE	29
POWER Files	34
Other Files	42
Sizing Optional Products	50
SETSDL Procedure	57
Chapter 4. Job Control Analysis and Conversion	59
Highlights	59
Job Control Scanner	60
Input from ICCF Library	60
Sample Jobs	60
Sample Output	61
Input from PL / SSL	62
Sample Jobs	62
Sample Output	64
Input from CMS	65
Sample EXEC / Job	65
Sample Output	67
Input from Unstructured Files	68
Sample Job	68
Sample Output	69
Job Control Exit	70

ICCF Procedures / Jobs	72
SUBMIT Procedure Sample	72
BACKUP/RESTORE of DTSFILE	76
Label Procedure Skeleton	77
ASI Master Procedure	80
Conditional Job Control	81
Return Code Assembler Subroutines	81
Sample PL/I Program	83
Sample COBOL Program	84
Link Procedures	85
Virtual I/O	85
VIO Implications	86
Usage of OPTION CATAL	87
Example of a LINKGO ICCF CLIST	88
Example of a LINKGO CMS EXEC	91
Output Produced by LINKGO CLIST	96
 Chapter 5. VSE Libraries Transfer	 97
Introduction	97
General Librarian Functions	98
Mapping of Old vs. New Librarian Functions	99
Old Librarian Functions	99
New Librarian Functions	100
Description of DASD Layouts	101
DASD Layout SIPO31	101
DASD Layout SYSWK1 VSE/SP 2.1	101
STDLABELS for SIPO31	102
STDLABELS for SYSWK1	102
Mapping of Libraries	103
Library Migration Graphic	103
Create Library Migration Table	104
Load Library Migration Table	105
Librarian Jobstreams	106
Library Creation (Definition)	106
Library Reorganization	109
Library Maintenance	142
Library Services	164
E-MACRO Maintenance and Services	186
Physical Library Migration	189
Old Libraries to New Libraries	189
Old Members to New Libraries	192
New Libraries to Old Libraries	194
Libraries in VSAM Managed Space	196
Definition	196
Extension	197
Librarian Migration Reference	198
Non-Migrated Functions	199
Job Control Functions	199
Librarian Services	203
Migrated Functions with Restrictions	220
Automatically Migrated Functions	224
Job Control Functions	224
Librarian Services	244

Chapter 6. User Files Transfer	307
Introduction	307
Migrate Databases	308
SQL Initialization	314
Chapter 7. ACF/VTAM Considerations	317
VTAM BOOKS	317
VTAM STARTUP BOOK	317
Configuration List	319
Other Books	320
Chapter 8. IBM Products Transfer	323
Migrate An Old Product	323
Sample Output of RETRACE PRODUCT	326
Chapter 9. Miscellaneous Jobs	329
Overview	329
FASTCOPY Samples	330
Backup/Restore in the Interactive Interface Environment	332
Backup Interactive Interface Files	333
Restore ICCF User Libraries	334
Restore of Interactive Interface VSAM Files	335
Create Standalone DUMP Program	336
Chapter 10. VM/VSE	337
Overview	337
VM/PASSTHRU	337
Sample Session Using VM/PASSTHRU	338
CMS EXEC Samples	345
Scanner Tape Creation	345
Assist in Migration of JCL	348
ROUTER Facility	348
How it Worked	348
ROUTER Replacement	350
Chapter 11. DASD Sharing	353
Functional Enhancements	353
Basic Functions	355
New IPL Parameters	355
Format of DLF Command	356
ASI Procedures	356
ASI Procedures CPU 1	356
IPL ASI Procedure	356
JCL ASI Procedure	358
ASI Procedures CPU 2	360
IPL ASI Procedure	360
JCL ASI Procedure	361
POWER Generation	362
System-Supplied Procedures	363
Structure	364
Clean-up	367
Summary of Cleaned-up Procedures	368
Procedures Have to Be Maintained	369

Samples of Cleaned-up Procedures	370
Prepare Procedures for DASD Sharing	375
LIBASGN Procedure	378
Invoking the LIBASGN Procedure	383
Two Interactive Interfaces - Procedure Modifications	384
DTRICCF Problem	384
VSAM Implications	390
Description	390
Logic Flow	391
Tailor ADM\$DTAB	392
Clean-up ICCF Startup Job	396
Two Interactive Interfaces - VSAM Files	398
Overview System File Sharability	399
Sharable Interactive Interface VSAM Files	400
Installation of Second Interactive Interface	401
VSAM Structure	402
VSAM Definitions	403
Loading of VSAM Files	406
Libraries in VSAM Managed Space	406
POWER Considerations	407
Setup of POWER Classes	407
Handling POWER Time Slices	409
 Appendix A. Table of Contents of Volume 1	 411
Index	417

FIGURES

1. Sample PASSTHRU EXEC	337
2. Overview	365

CHAPTER 1. INTRODUCTION

HOW TO USE THIS MANUAL

The book you are reading is a reference book, the descriptive part of the migration task is listed in Volume 1. Therefore, you find lots of samples in this volume, with very little accompanying text.

Use the index to find the samples more easily (look for the keywords such as MAINT, CSERV, and so on).

Some jobs are skeletons, that is, you must modify them or fill in the information specific for your system needs (for example, VOLID or FILE-ID); others are job samples, that is, you can use them without modification.

CHAPTER 2. INTERACTIVE INTERFACE

OVERVIEW

This chapter deals with the selection panel hierarchy of the VSE/SP 2.1 Interactive Interface for the:

- System Administrator (IESEADM)
- Application Programmer (IESEPROG)
- System Operator (IESEOPER)

The listings, that is, the summary of the respective paths are first listed in numerical, then in alphabetical sequence.

You also find examples for the migration of user-ids from existing systems.

SYSTEM ADMINISTRATION

Path Summary IESEADM

1	INSTALLATION
11	INSTALL PROGRAMS - (V2 FORMAT)
111	PREPARE FOR INSTALLATION
112	INSTALL PRODUCT(S) FROM TAPE
113	COMPLETE INSTALLATION
12	INSTALL PROGRAMS - (V1 FORMAT)
121	INSTALL PRODUCT(S) FROM TAPE(S)
122	COMPLETE INSTALLATION
13	INSTALL GENERATION FEATURE
14	IBM SERVICE
141	PRINT SERVICE DOCUMENTS
142	APPLY PTF
143	ALTER PHASE, MODULE, OR SOURCE
144	UNDO PHASE OR MODULE
145	INSTALL FAST SERVICE UPGRADE
146	RETRACE HISTORY FILE
147	REMOVE HISTORY RECORD
148	PERSONALIZE HISTORY FILE
15	INSTALL NETWORK TAPE
2	RESOURCE DEFINITION
21	USER INTERFACE TAILORING

211 MAINTAIN USER PROFILES
 212 MAINTAIN SELECTION PANELS
 213 MAINTAIN APPLICATION PROFILES
 22 FILE AND CATALOG MANAGEMENT
 23 DISPLAY VTOC
 24 LOCAL CONFIGURATION AND IPL
 241 CONFIGURE HARDWARE ADDRESSES
 242 CONFIGURE LOCAL NON-SNA 3270S
 243 CONFIGURE LOCAL SNA 3270S
 244 TAILOR IPL PROCEDURE
 245 MAINTAIN PRINTER FCB
 246 CATALOG PRINTER UCB
 25 REMOTE TERMINAL CONFIGURATION
 251 CONFIGURE ICA CONNECTED TERMINALS
 252 CONFIGURE NCP CONNECTED TERMINALS
 253 CONFIGURE EP CONNECTED TERMINALS
 254 CREATE NETWORK TAPE
 26 CHANGE VTAM APPLICATION NAME AND/OR SUBAREA NUMBER
 27 CREATE STARTUP BOOKS
 3 OPERATIONS
 31 SYSTEM CONSOLE
 32 MANAGE BATCH QUEUES
 33 DISPLAY ACTIVE USERS/SEND MESSAGE
 34 ENTER NEWS
 35 RETRIEVE MESSAGE
 36 SYSTEM STATUS
 361 DISPLAY SYSTEM ACTIVITY
 362 DISPLAY CHANNEL AND DEVICE ACTIVITY
 363 INVOKE CEMT
 37 BACKUP/RESTORE
 371 BACKUP/RESTORE VSAM OBJECTS
 3711 EXPORT VSAM FILE
 3712 IMPORT VSAM FILE
 3713 BACKUP VSAM FILE
 3714 RESTORE VSAM FILE
 3715 TRANSPORT CATALOG
 3716 COPY IN CATALOG
 3717 COPY OUT CATALOG
 372 BACKUP/RESTORE LIBRARY OBJECTS
 3721 BACKUP VSE LIBRARY ON TAPE
 3722 RESTORE VSE LIBRARY FROM TAPE
 3723 SCAN VSE LIBRARY BACKUP TAPE
 3724 BACKUP ICCF LIBRARY ON TAPE
 3725 RESTORE ICCF LIBRARY FROM TAPE
 373 BACKUP HISTORY FILE
 374 RESTORE HISTORY FILE
 375 BACKUP FROM DISK TO TAPE
 376 RESTORE FROM TAPE TO DISK
 377 COPY FROM ONE DISK TO ANOTHER
 38 PERSONAL COMPUTER MOVE UTILITIES
 39 TRANSFER FILES AND JOBS TO ANOTHER SYSTEM
 391 TRANSFER FILES TO ANOTHER SYSTEM
 392 RETRIEVE FILES FROM ANOTHER SYSTEM
 393 SUBMIT A JOB TO ANOTHER SYSTEM

4	PROBLEM HANDLING
41	ONLINE PROBLEM DETERMINATION
42	INSPECT MESSAGE LOG
43	STORAGE DUMP MANAGEMENT
44	INSPECT DUMP MANAGEMENT OUTPUT
5	PROGRAM DEVELOPMENT
51	PROGRAM DEVELOPMENT LIBRARY
52	CREATE APPLICATION JOBSTREAM
6	COMMAND MODE

Keyword Summary IESEADM

ACTIVE	33	DISPLAY ACTIVE USERS/SEND MESSAGE
ACTIVITY	361	DISPLAY SYSTEM ACTIVITY
"	362	DISPLAY CHANNEL AND DEVICE ACTIVITY
ADDRESSES	241	CONFIGURE HARDWARE ADDRESSES
ALTER	143	ALTER PHASE, MODULE, OR SOURCE
APPLICATION	213	MAINTAIN APPLICATION PROFILES
"	26	CHANGE VTAM APPLICATION NAME AND/OR SUBAREA NUMBER
"	52	CREATE APPLICATION JOBSTREAM
APPLY	142	APPLY PTF
BACKUP	37	BACKUP/RESTORE
"	371	BACKUP/RESTORE VSAM OBJECTS
"	3713	BACKUP VSAM FILE
"	372	BACKUP/RESTORE LIBRARY OBJECTS
"	3721	BACKUP VSE LIBRARY ON TAPE
"	3723	SCAN VSE LIBRARY BACKUP TAPE
"	3724	BACKUP ICCF LIBRARY ON TAPE
"	373	BACKUP HISTORY FILE
"	375	BACKUP FROM DISK TO TAPE
BATCH	32	MANAGE BATCH QUEUES
BOOKS	27	CREATE STARTUP BOOKS
CATALOG	22	FILE AND CATALOG MANAGEMENT
"	246	CATALOG PRINTER UCB
"	3715	TRANSPORT CATALOG
"	3716	COPY IN CATALOG
"	3717	COPY OUT CATALOG
CEMT	363	INVOKE CEMT
CHANGE	26	CHANGE VTAM APPLICATION NAME AND/OR SUBAREA NUMBER
CHANNEL	362	DISPLAY CHANNEL AND DEVICE ACTIVITY
COMMAND	6	COMMAND MODE
COMPLETE	113	COMPLETE INSTALLATION
"	122	COMPLETE INSTALLATION
COMPUTER	38	PERSONAL COMPUTER MOVE UTILITIES
CONFIGURATION	24	LOCAL CONFIGURATION AND IPL
"	25	REMOTE TERMINAL CONFIGURATION
CONFIGURE	241	CONFIGURE HARDWARE ADDRESSES
"	242	CONFIGURE LOCAL NON-SNA 3270S
"	243	CONFIGURE LOCAL SNA 3270S
"	251	CONFIGURE ICA CONNECTED TERMINALS
"	252	CONFIGURE NCP CONNECTED TERMINALS
"	253	CONFIGURE EP CONNECTED TERMINALS
CONNECTED	251	CONFIGURE ICA CONNECTED TERMINALS
"	252	CONFIGURE NCP CONNECTED TERMINALS
"	253	CONFIGURE EP CONNECTED TERMINALS
CONSOLE	31	SYSTEM CONSOLE

COPY	3716	COPY IN CATALOG
"	3717	COPY OUT CATALOG
"	377	COPY FROM ONE DISK TO ANOTHER
CREATE	254	CREATE NETWORK TAPE
"	27	CREATE STARTUP BOOKS
"	52	CREATE APPLICATION JOBSTREAM
DEFINITION	2	RESOURCE DEFINITION
DETERMINATION	41	ONLINE PROBLEM DETERMINATION
DEVELOPMENT	5	PROGRAM DEVELOPMENT
"	51	PROGRAM DEVELOPMENT LIBRARY
DEVICE	362	DISPLAY CHANNEL AND DEVICE ACTIVITY
DISK	375	BACKUP FROM DISK TO TAPE
"	376	RESTORE FROM TAPE TO DISK
"	377	COPY FROM ONE DISK TO ANOTHER
DISPLAY	23	DISPLAY VTOC
"	33	DISPLAY ACTIVE USERS/SEND MESSAGE
"	361	DISPLAY SYSTEM ACTIVITY
"	362	DISPLAY CHANNEL AND DEVICE ACTIVITY
DOCUMENTS	141	PRINT SERVICE DOCUMENTS
DUMP	43	STORAGE DUMP MANAGEMENT
"	44	INSPECT DUMP MANAGEMENT OUTPUT
ENTER	34	ENTER NEWS
EXPORT	3711	EXPORT VSAM FILE
FAST	145	INSTALL FAST SERVICE UPGRADE
FCB	245	MAINTAIN PRINTER FCB
FEATURE	13	INSTALL GENERATION FEATURE
FILE	146	RETRACE HISTORY FILE
"	148	PERSONALIZE HISTORY FILE
"	22	FILE AND CATALOG MANAGEMENT
"	3711	EXPORT VSAM FILE
"	3712	IMPORT VSAM FILE
"	3713	BACKUP VSAM FILE
"	3714	RESTORE VSAM FILE
"	373	BACKUP HISTORY FILE
"	374	RESTORE HISTORY FILE
FILES	39	TRANSFER FILES AND JOBS TO ANOTHER SYSTEM
"	391	TRANSFER FILES TO ANOTHER SYSTEM
"	392	RETRIEVE FILES FROM ANOTHER SYSTEM
FORMAT	11	INSTALL PROGRAMS - (V2 FORMAT)
"	12	INSTALL PROGRAMS - (V1 FORMAT)
GENERATION	13	INSTALL GENERATION FEATURE
HANDLING	4	PROBLEM HANDLING
HARDWARE	241	CONFIGURE HARDWARE ADDRESSES
HISTORY	146	RETRACE HISTORY FILE
"	147	REMOVE HISTORY RECORD
"	148	PERSONALIZE HISTORY FILE
"	373	BACKUP HISTORY FILE
"	374	RESTORE HISTORY FILE
ICA	251	CONFIGURE ICA CONNECTED TERMINALS

ICCF	3724	BACKUP ICCF LIBRARY ON TAPE
"	3725	RESTORE ICCF LIBRARY FROM TAPE
IMPORT	3712	IMPORT VSAM FILE
INSPECT	42	INSPECT MESSAGE LOG
"	44	INSPECT DUMP MANAGEMENT OUTPUT
INSTALL	11	INSTALL PROGRAMS - (V2 FORMAT)
"	112	INSTALL PRODUCT(S) FROM TAPE
"	12	INSTALL PROGRAMS - (V1 FORMAT)
"	121	INSTALL PRODUCT(S) FROM TAPE(S)
"	13	INSTALL GENERATION FEATURE
"	145	INSTALL FAST SERVICE UPGRADE
"	15	INSTALL NETWORK TAPE
INSTALLATION	1	INSTALLATION
"	111	PREPARE FOR INSTALLATION
"	113	COMPLETE INSTALLATION
"	122	COMPLETE INSTALLATION
INTERFACE	21	USER INTERFACE TAILORING
INVOKE	363	INVOKE CEMT
IPL	24	LOCAL CONFIGURATION AND IPL
"	244	TAILOR IPL PROCEDURE
JOB	393	SUBMIT A JOB TO ANOTHER SYSTEM
JOBS	39	TRANSFER FILES AND JOBS TO ANOTHER SYSTEM
JOBSTREAM	52	CREATE APPLICATION JOBSTREAM
LIBRARY	372	BACKUP/RESTORE LIBRARY OBJECTS
"	3721	BACKUP VSE LIBRARY ON TAPE
"	3722	RESTORE VSE LIBRARY FROM TAPE
"	3723	SCAN VSE LIBRARY BACKUP TAPE
"	3724	BACKUP ICCF LIBRARY ON TAPE
"	3725	RESTORE ICCF LIBRARY FROM TAPE
"	51	PROGRAM DEVELOPMENT LIBRARY
LOCAL	24	LOCAL CONFIGURATION AND IPL
"	242	CONFIGURE LOCAL NON-SNA 3270S
"	243	CONFIGURE LOCAL SNA 3270S
LOG	42	INSPECT MESSAGE LOG
MAINTAIN	211	MAINTAIN USER PROFILES
"	212	MAINTAIN SELECTION PANELS
"	213	MAINTAIN APPLICATION PROFILES
"	245	MAINTAIN PRINTER FCB
MANAGE	32	MANAGE BATCH QUEUES
MANAGEMENT	22	FILE AND CATALOG MANAGEMENT
"	43	STORAGE DUMP MANAGEMENT
"	44	INSPECT DUMP MANAGEMENT OUTPUT
MESSAGE	33	DISPLAY ACTIVE USERS/SEND MESSAGE
"	35	RETRIEVE MESSAGE
"	42	INSPECT MESSAGE LOG
MODE	6	COMMAND MODE
MODULE	143	ALTER PHASE, MODULE, OR SOURCE
"	144	UNDO PHASE OR MODULE
MOVE	38	PERSONAL COMPUTER MOVE UTILITIES

NAME	26	CHANGE VTAM APPLICATION NAME AND/OR SUBAREA NUMBER
NCP	252	CONFIGURE NCP CONNECTED TERMINALS
NETWORK	15	INSTALL NETWORK TAPE
"	254	CREATE NETWORK TAPE
NEWS	34	ENTER NEWS
NUMBER	26	CHANGE VTAM APPLICATION NAME AND/OR SUBAREA NUMBER
OBJECTS	371	BACKUP/RESTORE VSAM OBJECTS
"	372	BACKUP/RESTORE LIBRARY OBJECTS
ONLINE	41	ONLINE PROBLEM DETERMINATION
OPERATIONS	3	OPERATIONS
OUTPUT	44	INSPECT DUMP MANAGEMENT OUTPUT
PANELS	212	MAINTAIN SELECTION PANELS
PERSONAL	38	PERSONAL COMPUTER MOVE UTILITIES
PERSONALIZE	148	PERSONALIZE HISTORY FILE
PHASE	143	ALTER PHASE, MODULE, OR SOURCE
"	144	UNDO PHASE OR MODULE
PREPARE	111	PREPARE FOR INSTALLATION
PRINT	141	PRINT SERVICE DOCUMENTS
PRINTER	245	MAINTAIN PRINTER FCB
"	246	CATALOG PRINTER UCB
PROBLEM	4	PROBLEM HANDLING
"	41	ONLINE PROBLEM DETERMINATION
PROCEDURE	244	TAILOR IPL PROCEDURE
PRODUCT	112	INSTALL PRODUCT(S) FROM TAPE
"	121	INSTALL PRODUCT(S) FROM TAPE(S)
PROFILES	211	MAINTAIN USER PROFILES
"	213	MAINTAIN APPLICATION PROFILES
PROGRAM	5	PROGRAM DEVELOPMENT
"	51	PROGRAM DEVELOPMENT LIBRARY
PROGRAMS	11	INSTALL PROGRAMS - (V2 FORMAT)
"	12	INSTALL PROGRAMS - (V1 FORMAT)
PTF	142	APPLY PTF
QUEUES	32	MANAGE BATCH QUEUES
RECORD	147	REMOVE HISTORY RECORD
REMOTE	25	REMOTE TERMINAL CONFIGURATION
REMOVE	147	REMOVE HISTORY RECORD
RESOURCE	2	RESOURCE DEFINITION
RESTORE	37	BACKUP/RESTORE
"	371	BACKUP/RESTORE VSAM OBJECTS
"	3714	RESTORE VSAM FILE
"	372	BACKUP/RESTORE LIBRARY OBJECTS
"	3722	RESTORE VSE LIBRARY FROM TAPE
"	3725	RESTORE ICCF LIBRARY FROM TAPE
"	374	RESTORE HISTORY FILE
"	376	RESTORE FROM TAPE TO DISK
RETRACE	146	RETRACE HISTORY FILE
RETRIEVE	35	RETRIEVE MESSAGE
"	392	RETRIEVE FILES FROM ANOTHER SYSTEM
SCAN	3723	SCAN VSE LIBRARY BACKUP TAPE

SELECTION	212	MAINTAIN SELECTION PANELS
SEND	33	DISPLAY ACTIVE USERS/SEND MESSAGE
SERVICE	14	IBM SERVICE
"	141	PRINT SERVICE DOCUMENTS
"	145	INSTALL FAST SERVICE UPGRADE
SNA	242	CONFIGURE LOCAL NON-SNA 3270S
"	243	CONFIGURE LOCAL SNA 3270S
SOURCE	143	ALTER PHASE, MODULE, OR SOURCE
STARTUP	27	CREATE STARTUP BOOKS
STATUS	36	SYSTEM STATUS
STORAGE	43	STORAGE DUMP MANAGEMENT
SUBAREA	26	CHANGE VTAM APPLICATION NAME AND/OR SUBAREA NUMBER
SUBMIT	393	SUBMIT A JOB TO ANOTHER SYSTEM
SYSTEM	31	SYSTEM CONSOLE
"	36	SYSTEM STATUS
"	361	DISPLAY SYSTEM ACTIVITY
"	39	TRANSFER FILES AND JOBS TO ANOTHER SYSTEM
"	391	TRANSFER FILES TO ANOTHER SYSTEM
"	392	RETRIEVE FILES FROM ANOTHER SYSTEM
"	393	SUBMIT A JOB TO ANOTHER SYSTEM
TAILOR	244	TAILOR IPL PROCEDURE
TAILORING	21	USER INTERFACE TAILORING
TAPE	112	INSTALL PRODUCT(S) FROM TAPE
"	121	INSTALL PRODUCT(S) FROM TAPE(S)
"	15	INSTALL NETWORK TAPE
"	254	CREATE NETWORK TAPE
"	3721	BACKUP VSE LIBRARY ON TAPE
"	3722	RESTORE VSE LIBRARY FROM TAPE
"	3723	SCAN VSE LIBRARY BACKUP TAPE
"	3724	BACKUP ICCF LIBRARY ON TAPE
"	3725	RESTORE ICCF LIBRARY FROM TAPE
"	375	BACKUP FROM DISK TO TAPE
"	376	RESTORE FROM TAPE TO DISK
TERMINAL	25	REMOTE TERMINAL CONFIGURATION
TERMINALS	251	CONFIGURE ICA CONNECTED TERMINALS
"	252	CONFIGURE NCP CONNECTED TERMINALS
"	253	CONFIGURE EP CONNECTED TERMINALS
TRANSFER	39	TRANSFER FILES AND JOBS TO ANOTHER SYSTEM
"	391	TRANSFER FILES TO ANOTHER SYSTEM
TRANSPORT	3715	TRANSPORT CATALOG
UCB	246	CATALOG PRINTER UCB
UNDO	144	UNDO PHASE OR MODULE
UPGRADE	145	INSTALL FAST SERVICE UPGRADE
USER	21	USER INTERFACE TAILORING
"	211	MAINTAIN USER PROFILES
USERS	33	DISPLAY ACTIVE USERS/SEND MESSAGE
UTILITIES	38	PERSONAL COMPUTER MOVE UTILITIES

VSAM	371	BACKUP/RESTORE VSAM OBJECTS
"	3711	EXPORT VSAM FILE
"	3712	IMPORT VSAM FILE
"	3713	BACKUP VSAM FILE
"	3714	RESTORE VSAM FILE
VSE	3721	BACKUP VSE LIBRARY ON TAPE
"	3722	RESTORE VSE LIBRARY FROM TAPE
"	3723	SCAN VSE LIBRARY BACKUP TAPE
VTAM	26	CHANGE VTAM APPLICATION NAME AND/OR SUBAREA NUMBER
VTOC	23	DISPLAY VTOC
3270S	242	CONFIGURE LOCAL NON-SNA 3270S
"	243	CONFIGURE LOCAL SNA 3270S

APPLICATION DEVELOPMENT

Path Summary IESEPROG

- 1 PROGRAM DEVELOPMENT LIBRARY
- 2 MANAGE BATCH QUEUES
- 3 CREATE APPLICATION JOBSTREAM
- 4 PROBLEM HANDLING
- 41 ONLINE PROBLEM DETERMINATION
- 42 INSPECT MESSAGE LOG
- 43 STORAGE DUMP MANAGEMENT
- 44 INSPECT DUMP MANAGEMENT OUTPUT
- 5 OPERATIONS
- 51 SYSTEM CONSOLE
- 52 DISPLAY ACTIVE USERS/SEND MESSAGE
- 53 ENTER NEWS
- 54 RETRIEVE MESSAGES
- 55 DISPLAY SYSTEM ACTIVITY
- 56 INVOKE CEMT
- 57 BACKUP/RESTORE
- 571 EXPORT VSAM FILE
- 572 IMPORT VSAM FILE
- 573 BACKUP VSAM FILE
- 574 RESTORE VSAM FILE
- 575 BACKUP ICCF LIBRARY ON TAPE
- 576 RESTORE ICCF LIBRARY FROM TAPE
- 577 BACKUP FROM DISK TO TAPE
- 578 RESTORE FROM TAPE TO DISK
- 579 COPY FROM ONE DISK TO ANOTHER
- 6 FILE MANAGEMENT
- 7 COMMAND MODE

Keyword Summary IESEPROG

ACTIVE	52	DISPLAY ACTIVE USERS/SEND MESSAGE
ACTIVITY	55	DISPLAY SYSTEM ACTIVITY
APPLICATION	3	CREATE APPLICATION JOBSTREAM
BACKUP	57	BACKUP/RESTORE
"	573	BACKUP VSAM FILE
"	575	BACKUP ICCF LIBRARY ON TAPE
"	577	BACKUP FROM DISK TO TAPE
BATCH	2	MANAGE BATCH QUEUES
CEMT	56	INVOKE CEMT
COMMAND	7	COMMAND MODE
CONSOLE	51	SYSTEM CONSOLE
COPY	579	COPY FROM ONE DISK TO ANOTHER
CREATE	3	CREATE APPLICATION JOBSTREAM
DETERMINATION	41	ONLINE PROBLEM DETERMINATION
DEVELOPMENT	1	PROGRAM DEVELOPMENT LIBRARY
DISK	577	BACKUP FROM DISK TO TAPE
"	578	RESTORE FROM TAPE TO DISK
"	579	COPY FROM ONE DISK TO ANOTHER
DISPLAY	52	DISPLAY ACTIVE USERS/SEND MESSAGE
"	55	DISPLAY SYSTEM ACTIVITY
DUMP	43	STORAGE DUMP MANAGEMENT
"	44	INSPECT DUMP MANAGEMENT OUTPUT
ENTER	53	ENTER NEWS
EXPORT	571	EXPORT VSAM FILE
FILE	571	EXPORT VSAM FILE
"	572	IMPORT VSAM FILE
"	573	BACKUP VSAM FILE
"	574	RESTORE VSAM FILE
"	6	FILE MANAGEMENT
HANDLING	4	PROBLEM HANDLING
ICCF	575	BACKUP ICCF LIBRARY ON TAPE
"	576	RESTORE ICCF LIBRARY FROM TAPE
IMPORT	572	IMPORT VSAM FILE
INSPECT	42	INSPECT MESSAGE LOG
"	44	INSPECT DUMP MANAGEMENT OUTPUT
INVOKE	56	INVOKE CEMT
JOBSTREAM	3	CREATE APPLICATION JOBSTREAM
LIBRARY	1	PROGRAM DEVELOPMENT LIBRARY
"	575	BACKUP ICCF LIBRARY ON TAPE
"	576	RESTORE ICCF LIBRARY FROM TAPE
LOG	42	INSPECT MESSAGE LOG
MANAGE	2	MANAGE BATCH QUEUES
MANAGEMENT	43	STORAGE DUMP MANAGEMENT
"	44	INSPECT DUMP MANAGEMENT OUTPUT
"	6	FILE MANAGEMENT
MESSAGE	42	INSPECT MESSAGE LOG
"	52	DISPLAY ACTIVE USERS/SEND MESSAGE
MESSAGES	54	RETRIEVE MESSAGES
MODE	7	COMMAND MODE
NEWS	53	ENTER NEWS
ONLINE	41	ONLINE PROBLEM DETERMINATION
OPERATIONS	5	OPERATIONS

OUTPUT	44	INSPECT DUMP MANAGEMENT OUTPUT
PROBLEM	4	PROBLEM HANDLING
"	41	ONLINE PROBLEM DETERMINATION
PROGRAM	1	PROGRAM DEVELOPMENT LIBRARY
QUEUES	2	MANAGE BATCH QUEUES
RESTORE	57	BACKUP/RESTORE
"	574	RESTORE VSAM FILE
"	576	RESTORE ICCF LIBRARY FROM TAPE
"	578	RESTORE FROM TAPE TO DISK
RETRIEVE	54	RETRIEVE MESSAGES
SEND	52	DISPLAY ACTIVE USERS/SEND MESSAGE
STORAGE	43	STORAGE DUMP MANAGEMENT
SYSTEM	51	SYSTEM CONSOLE
"	55	DISPLAY SYSTEM ACTIVITY
TAPE	575	BACKUP ICCF LIBRARY ON TAPE
"	576	RESTORE ICCF LIBRARY FROM TAPE
"	577	BACKUP FROM DISK TO TAPE
"	578	RESTORE FROM TAPE TO DISK
USERS	52	DISPLAY ACTIVE USERS/SEND MESSAGE
VSAM	571	EXPORT VSAM FILE
"	572	IMPORT VSAM FILE
"	573	BACKUP VSAM FILE
"	574	RESTORE VSAM FILE

SYSTEM OPERATION

Path Summary IESEOPER

- 1 PROGRAM DEVELOPMENT LIBRARY
- 2 SYSTEM CONSOLE
- 3 MANAGE BATCH QUEUES
- 4 DISPLAY ACTIVE USERS/SEND MESSAGE
- 5 ENTER NEWS
- 6 RETRIEVE MESSAGE
- 7 DISPLAY SYSTEM ACTIVITY

Keyword Summary IESEOPER

ACTIVE	4	DISPLAY ACTIVE USERS/SEND MESSAGE
ACTIVITY	7	DISPLAY SYSTEM ACTIVITY
BATCH	3	MANAGE BATCH QUEUES
CONSOLE	2	SYSTEM CONSOLE
DEVELOPMENT	1	PROGRAM DEVELOPMENT LIBRARY
DISPLAY	4	DISPLAY ACTIVE USERS/SEND MESSAGE
"	7	DISPLAY SYSTEM ACTIVITY
ENTER	5	ENTER NEWS
LIBRARY	1	PROGRAM DEVELOPMENT LIBRARY
MANAGE	3	MANAGE BATCH QUEUES
MESSAGE	4	DISPLAY ACTIVE USERS/SEND MESSAGE
"	6	RETRIEVE MESSAGE
NEWS	5	ENTER NEWS
PROGRAM	1	PROGRAM DEVELOPMENT LIBRARY
QUEUES	3	MANAGE BATCH QUEUES
RETRIEVE	6	RETRIEVE MESSAGE
SEND	4	DISPLAY ACTIVE USERS/SEND MESSAGE
SYSTEM	2	SYSTEM CONSOLE
"	7	DISPLAY SYSTEM ACTIVITY
USERS	4	DISPLAY ACTIVE USERS/SEND MESSAGE

Migrate VSE/ICCF User Profiles

The following is an example of a job that creates user profiles from a backup copy of the **DTSFILE** using the VSE/SP-provided user profiles 'SYSA' and 'PROG' as models.

This sample job stream is included in VSE/ICCF library 59 as member IESBLDUP and may be tailored to suit your needs.

```
* $$ JOB  JNM=IESBLDUP,CLASS=A,DISP=D
// JOB IESBLDUP MIGRATE USER PROFILES
// ASSGN SYS010,DISK,TEMP,VOL=SYSWK1,SHR
// TLBL DTSRSTR,,,ICCF01
// ASSGN SYS004,X'1B0'
// PAUSE UPDATE=YES IS SPECIFIED, CLOSE IESCNTRL BEFORE PROCEEDING
// EXEC IESBLDUP,SIZE=32K
DTSFILE=NO,DTSRSTR=YES,SNT=NO,UPDATE=YES,ADMN=SYSA,PROG=PROG
* $$ SLI  ICCF=(ADM$USRT),LIB=50
/*
/&
* $$ EOJ
```

This example edits input from the DTSRSTR file, CICS/VS SNT, and user provided ADD statements. After is edit is made, you may change the UPDATE parameter to 'YES' to actually perform the user profile adds. The example assumes the user profiles 'SYSA', 'PROG', 'GENL', 'USR1', and 'USR2' are defined on the VSE/SP control file prior to running the program. 'SYSA' and 'PROG' are distributed with VSE/SP. 'GENL', 'USR1', and 'USR2' have to be defined by you.

```
* $$ JOB JNM=IESBLD01,CLASS=A,DISP=D
// JOB IESBLD01 MIGRATE USER PROFILES
// ASSGN SYS010,DISK,TEMP,VOL=SYSWK1,SHR
// TLBL DTSRSTR,,ICCF01
// ASSGN SYS004,X'1B0'
// PAUSE IESCNTRL MUST BE CLOSED IF UPDATE=YES IS SPECIFIED
// EXEC IESBLDUP,SIZE=32K
DTSFILE=NO,DTSRSTR=YES,SNT=YES,UPDATE=NO,ADMN=SYSA,GENL=GENL
JOES,CISCO,20,USR1
JIMS,MONTY,21,USR1
RAYS,LOST,22,USR2
FISH,HOWARD,23,SYSA
BULL,WAPITI,24,PROG
SPOT,SECRET,25,JIMS
* $$ SLI ICCF=(ADM$USRT),LIB=50
/*
/&
* $$ EOJ
```


The following example creates user profiles from the VSE/ICCF DTSFILE using the user-defined user profiles 'USRA', 'USRB', and 'USRC' as models. THE CICS/VS SNT is loaded and processed, DTSRSTR is not processed. The VSE/SP control file is updated with the new profiles and job DTRMIGR to update the DTSFILE is put into the POWER RDR queue.

```
* $$ JOB JNM=IESBLD02,CLASS=A,DISP=D
// JOB IESBLD02 MIGRATE USER PROFILES
// ASSGN SYS010,DISK,TEMP,VOL=SYSWK1,SHR
// PAUSE UPDATE=YES IS SPECIFIED, CLOSE IESCNTRL BEFORE PROCEEDING
// EXEC IESBLDUP,SIZE=32K
DTSFILE=YES,DTSRSTR=NO,SNT=YES,UPDATE=YES
ADMN=USRA,PROG=USRB,GENL=USRC
* $$ SLI ICCF=(ADM$USRT),LIB=50
/*
/&
* $$ EOJ
```


CHAPTER 3. SYSTEM ORGANIZATION

HIGHLIGHTS

This chapter deals with the regeneration of some system components as well as the relocation of certain system files and the determination of space needed for various products.

A summary is given below:

- A sample of how to generate your own tailored version of ICCF (see also job skeleton SKICFGEN in ICCF library 59).
- A sample of how to tailor POWER including Shared Spooling (see job skeleton SKPWRGEN in library 59).
- Hints and samples on how to extend and relocate various system files such as:
 - Libraries
 - ICCF and POWER files
 - Other system files
- A method (as provided by the Interactive Interface) to determine the library space required by optional products.
- A sample program that loads the most important phases into the SVA.

ICCF GENERATION

```
* $$ JOB JNM=ICCFGEN,CLASS=0,DISP=D,NTFY=YES
* $$ LST DISP=D,CLASS=R,DEST=(*,MERZ)
// JOB ICCFGEN TAILOR ICCF FOR VAE ENVIRONMENT
// OPTION NODUMP
ON $RC > 0 GOTO $EOJ
// LIBDEF *,SEARCH=IJSYSRS.SYSLIB,CATALOG=IJSYSRS.SYSLIB
// OPTION CATAL,NODECK
    PHASE DTSIGEN,*
// EXEC ASSEMBLY
    TITLE 'DTSIGEN -- TAILORED VERSION OF ICCF'
ICCFCPU1 DTSOPTNS ALTSEC=NO,
        ATN2741=NO,           WAS 'YES' IN SP VERSION
        CANKEY=PA2,
        CISIZE=2048,
```

C
M
C
C

POWER Generation

```

* $$ JOB JNM=POWERGEN,CLASS=0,DISP=D
* $$ LST CLASS=R,DISP=D,DEST=(*,MERZ)
// JOB POWERGEN ASSEMBLE NEW POWER FOR SHARED ENVIRONMENT
LIBDEF PHASE,CATALOG=PRD2.CONFIG
// OPTION CATAL,NODECK
// EXEC ASSEMBLY
PWR      TITLE 'VSE/POWER - PWRSHRSP GENERATION '
          PRINT GEN
          SPACE 3
PWRSHRD  POWER
          ACCOUNT=YES,
          BLOCKGP=80,           WAS '0'
          CLRPRT=YES,
          COPYSEP=NO,           WAS 'YES'
          DBLK=4080,            WAS '0'
          FEED=NO,
          JLOG=YES,
          JSEP=(0,0),
          LTAB=(10,00,05,10,15,20,25,30,35,40,45,50,56),
          MEMENTYPE=SOURCE,     NEW
          MRKFRM=YES,
          MULT12=NO,
          NTFYMSG=100,
          PAUSE=NO,
          PNET=PNDT,            NEW
          PRI=3,
          RBS=(0,0),
          RDREXIT=NO,
          RJEBSC=NO,            NEW
          SHARED=(Q,A),         ** SHARED SPOOLING **
          SPOOL=YES,
          STDCARD=(0,0),
          STDLINE=(0,0),
          SUBLIB=P,
          SYSID=1,               ** SHARED SPOOLING **
          TIME=(5,1,60),        ** SHARED SPOOLING **
          TRACKGP=0
          END
/*
// EXEC LNKEDT,PARM='MSHP'
/&
* $$ EOJ

```

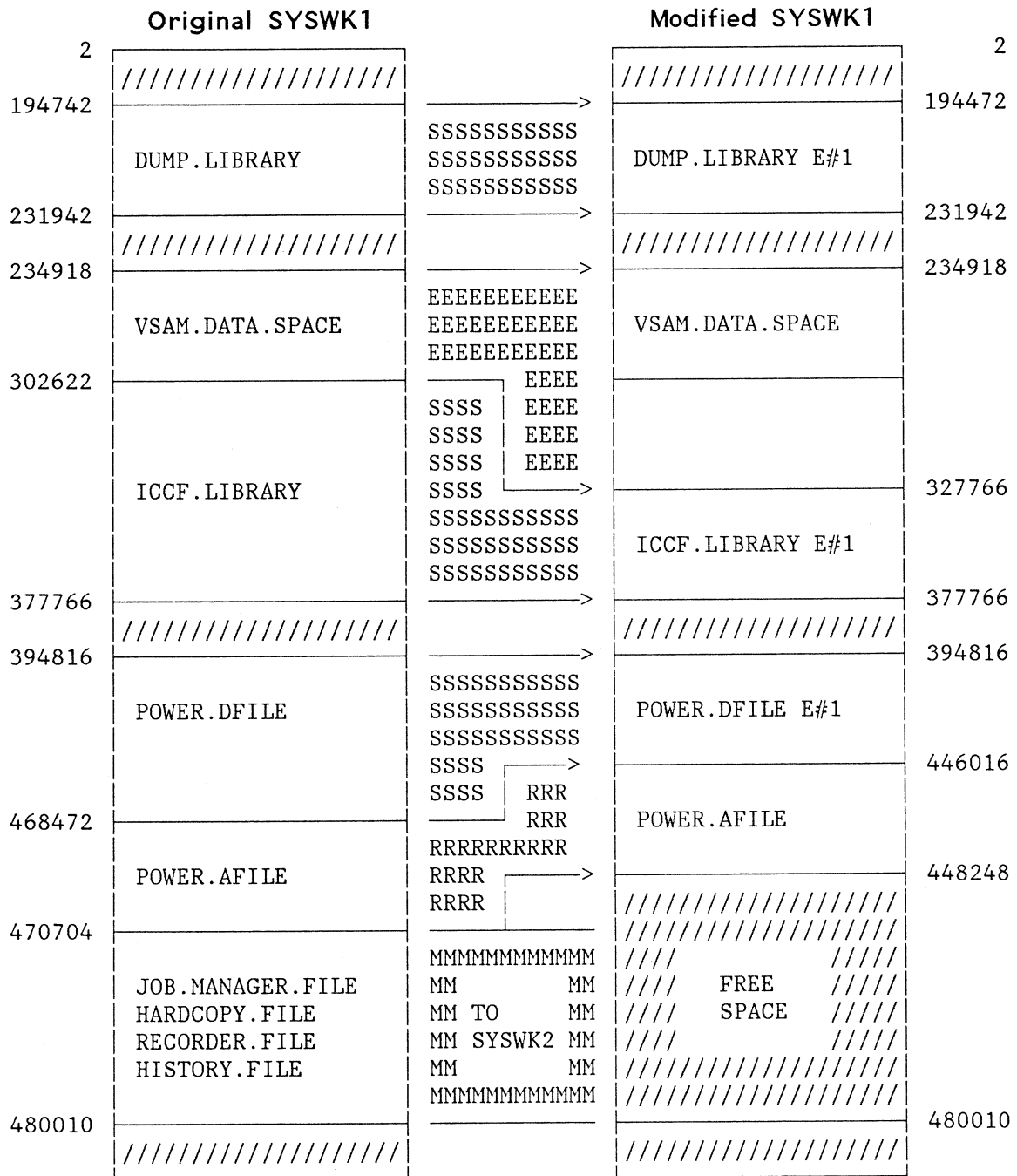
C
C
M
C
M
M
C
C
C
A
C
C
C
A
C
A
M
C
C
C
A
A

Tailor Libraries / Files

Highlights

- Extend Libraries in BAM space
- Extend Libraries in VSAM managed space
- Extend and split the ICCF DTSFILE
- Extend and split the POWER IJAFILE
- Move the System History File
- Move the Job Manager File
- Extend and split the DUMP Library

Note: The following graphics and job samples are based on the layout on an IBM 3370-1.



Legend: S=Split R=Relocated
 E=Extended M=Moved

Layout SYSWK2

			2
Moved from SYSWK1	—————>	VSESP.JOB.MANAGER.FILE	
			2048
		////////////////////////////////	2176
Moved from SYSWK1	—————	VSE.SYSTEM.HISTORY.FILE	
			6272
Extension of SYSWK1	└—————>	VSE.DUMP.LIBRARY E#2	
			43472
		////////////////////////////////	155000
Extension of SYSWK1	└—————>	VSAM.DATA.SPACE	
			247814
		////////////////////////////////	254000
Extension of SYSWK1	└—————>	ICCF.LIBRARY E#2	
			304000
Extension of SYSWK1	└—————>	VSE.POWER.DATA.FILE E#2	
			355200
Moved from SYSWK1	—————>	VSE.HARDCOPY.FILE	
			360194
Moved from SYSWK1	—————>	VSE.RECORDER.FILE	
			360946
		////////////////////////////////	

LIBRARIES

In BAM Space

- BACKUP the Library which has to be extended
- Deactivate the Library
 - Method 1: LIBDROP the Library in all partitions that have a permanent LIBDEF to it
 - Method 2: UNBATCH all partitions that have a permanent LIBDEF to this Library
- RESTORE the Library supplying new EXTENT information
 - The necessary DEFINES will be made automatically
- Update procedure STDLABEL with the new label information
- Reactivate the Library by either restarting the partition, or by establishing the appropriate LIBDEFs

In VSAM Managed Space

- Extending libraries in VSAM managed space is very simple and straightforward
 - DEFINE additional VSAM space (in multiples of the secondary allocation of the Library)
 - ALTER the cluster for this Library to include the new space
- The following sample has been created using the dialogues of the Interactive Interface

```
* $$ JOB JNM=VSAMEXT,CLASS=0,DISP=D,NTFY=YES
// JOB VSAMEXT DEFINE SPACE
// EXEC IDCAMS,SIZE=AUTO
DEFINE SPACE (
    BLOCKS (46000 )
    ORIGIN (44240 )
    VOLUME (SYSWK2))
    CATALOG (VSAM.MASTER.CATALOG)
IF LASTCC NE 0 THEN CANCEL JOB
ALTER VSE.PR2.LIBRARY.DATA
    ADDVOLUMES (SYSWK2)
    CATALOG (VSAM.MASTER.CATALOG)
/*
/&
* $$ EOJ
```

ICCF DTSFILE

Summary of Jobs

TAILIC00	Backup the existing DTSFILE to tape
TAILIC10	Store the new DTRICCF procedure into an auxiliary sublibrary (see Note)
TAILIC20	Create the new STDLABEL procedure and store it into an auxiliary sublibrary
TAILIC30	Format the new DTSFILE and create the appropriate ICCF libraries
TAILIC40	Restore the DTSFILE into the new extents
TAILIC50	Copy the new procedures from the auxiliary sublibrary to IJSYSRS.SYSLIB and PRD2.SAVE Shut down the system and re-IPL.

```

* $$ JOB JNM=TAILIC00,CLASS=0,DISP=L
// JOB TAILIC00  BACKUP THE EXISTING ICCF LIBRARY
// TLBL DTSBKUP,'ICCF BACKUP'
// ASSGN SYS005,1B2,D0
// EXEC PROC=DTRICCF
// EXEC DTSUTIL
BACKUP
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=TAILIC10,CLASS=0,DISP=L
// JOB TAILIC10  UPDATE DTRICCF PROCEDURE
// EXEC LIBR,PARM='MSHP'
ACCESS S=PRD2.TAILOR
CATALOG DTRICCF.PROC      DATA=YES REPLACE=YES
// ASSGN SYS010,DISK,VOL=SYSWK1,SHR      DTSFILE EXTENT 1
// ASSGN SYS011,DISK,VOL=SYSWK2,SHR      DTSFILE EXTENT 2
/+
/&
* $$ EOJ

```

```

* $$ JOB JNM=TAILIC20,CLASS=0,DISP=L
// JOB TAILIC20 UPDATE STDLABEL PROCEDURE
// EXEC LIBR,PARM='MSHP'
ACCESS S=PRD2.TAILOR
CATALOG STDLABEL.PROC          DATA=YES REPLACE=YES
// OPTION STDLABEL
/. STANDARD LABELS FOR 3370-1
// DLBL IJSYSRS,'VSE.SYSRES.LIBRARY',1999/365,SD
// EXTENT SYSRES,DOSRES,1,0,2,55116
.
.
.
// DLBL DTSFILE,'ICCF.LIBRARY',1999/365,DA
// EXTENT SYS010,SYSWK1,1,0,327766,50000
// EXTENT SYS011,SYSWK2,1,1,254000,50000
.
.
.
// EXEC PROC=STDLABUP - INCLUDE VSAM FULIST FILE LABELS
// EXEC PROC=STDLABUS - INCLUDE USER STANDARD AND PARSTD LABELS
// OPTION USRLABEL - TO CLEAR LABEL AREA AND RESET LOCK
/*
/+
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=TAILIC30,CLASS=0,DISP=L
* $$ LST  DISP=D,CLASS=A,DEST=(*,0)
* $$ PUN  DISP=D,CLASS=A,RBM=(1000),DEST=(*,0)
// JOB TAILIC30  FORMAT NEW ICCF LIBRARY
// DLBL DTSFILE,'ICCF.LIBRARY',1999/365,DA
// EXTENT SYS010,SYSWK1,1,0,327766,50000
// EXTENT SYS011,SYSWK2,1,1,254000,50000
// ASSGN SYS010,DISK,VOL=SYSWK1,SHR      DTSFILE EXTENT 1
// ASSGN SYS011,DISK,VOL=SYSWK2,SHR      DTSFILE EXTENT 2
// EXEC DTSUTIL
FORMAT USERS(99),LIBRARIES(99)
*  ADD LIBRARY 1 WHICH BELONGS TO SYSTEM
ADD LIBRARY FREESPACE(40) DATE
*  ADD LIBRARY 2 WHICH WILL BE USED AS THE COMMON LIBRARY
ADD LIBRARY FREESPACE(10) DATE
*  ADD LIBRARIES 3,4,5, AND 6 WHICH WILL BE USED AS PUBLIC LIBRARIES
ADD LIBRARY MAXDIR(200) FREESPACE(25) DATE NOCOMMON PUBLIC
ADD LIBRARY MAXDIR(200) FREESPACE(25) DATE NOCOMMON PUBLIC
ADD LIBRARY MAXDIR(200) FREESPACE(25) DATE NOCOMMON PUBLIC
ADD LIBRARY MAXDIR(200) FREESPACE(25) DATE NOCOMMON PUBLIC
*  ADD LIBRARIES 7 THRU 49 WHICH WILL BE USED AS USER PRIVATE LIBRARIES
ADD LIBRARY MAXDIR(200) FREESPACE(25) DATE
ADD LIBRARY MAXDIR(200) FREESPACE(25) DATE
.
.
ADD LIBRARY MAXDIR(200) FREESPACE(25) DATE
ADD LIBRARY MAXDIR(200) FREESPACE(25) DATE
*  ADD LIBRARIES 50 THRU 69 WHICH WILL BE USED FOR IPF LIBRARIES
ADD LIBRARY DATE NOCOMMON PUBLIC
ADD LIBRARY DATE NOCOMMON PUBLIC
.
.
ADD LIBRARY DATE NOCOMMON PUBLIC
ADD LIBRARY DATE NOCOMMON PUBLIC
*  ADD LIBRARIES 70 THROUGH 99.
ADD LIBRARY MAXDIR(200) FREESPACE(25) DATE
ADD LIBRARY MAXDIR(200) FREESPACE(25) DATE
.
.
ADD LIBRARY MAXDIR(200) FREESPACE(25) DATE
ADD LIBRARY MAXDIR(200) FREESPACE(25) DATE
END
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=TAILIC40,CLASS=0,DISP=L
* $$ LST  DISP=D,CLASS=A,DEST=(*,0)
* $$ PUN  DISP=D,CLASS=A,RBM=(1000),DEST=(*,0)
// JOB TAILIC40  RESTORE ICCF LIBRARY
// DLBL DTSFILE,'ICCF.LIBRARY',1999/365,DA
// EXTENT SYS010,SYSWK1,1,0,327766,50000
// EXTENT SYS011,SYSWK2,1,1,254000,50000
// ASSGN SYS010,DISK,VOL=SYSWK1,SHR      DTSFILE EXTENT 1
// ASSGN SYS011,DISK,VOL=SYSWK2,SHR      DTSFILE EXTENT 2
// PAUSE  ANYTHING TO BE CHANGED
// TLBL DTSRSTR,'ICCF BACKUP'
// ASSGN SYS004,1B2
// EXEC DTSUTIL
RESTORE ALL
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=TAILIC50,CLASS=0,DISP=L
* $$ LST  DISP=D,CLASS=A,DEST=(*,0)
* $$ PUN  DISP=D,CLASS=A,RBM=(1000),DEST=(*,0)
// JOB TAILIC50  COPY PROCEDURES STDLABEL AND DTRICCF
* TO IJSYSRS.SYSLIB AND PRD2.SAVE
// EXEC LIBR,PARM='MSHP'
CONNECT S=PRD2.TAILOR:PRD2.SAVE
COPY DTRICCF,STDLABEL. R=Y
CONNECT S=PRD2.TAILOR:IJSYSRS.SYSLIB
COPY DTRICCF,PROC,STDLABEL. R=Y
/*
/&
* $$ EOJ

```

POWER FILES

Summary of Jobs

TAILPW00	Backup the existing POWER Queues by means of POFFLOAD
TAILPW10	Generate the new POWER
TAILPW20	Save original procedures from IJSYSRS.SYSLIB in an auxiliary sublibrary
TAILPW30	Update procedure DTRPOWR to reflect the correct assignments of the new POWER files (see Note)
TAILPW40	Store the updated STDLABEL procedure in an auxiliary sublibrary
TAILPW50	Catalog an auxiliary startup procedure for POWER with FORMAT=D,A
TAILPW60	Copy the new procedures from the auxiliary sublibrary to IJSYSRS.SYSLIB and PRD2.SAVE
TAILPW70	Shut down the system and re-IPL
TAILPW80	Catalog the final POWER startup procedure (with FORMAT=NO) into IJSYSRS.SYSLIB and PRD2.SAVE


```
* $$ JOB JNM=TAILPW00,CLASS=0,DISP=L
// JOB TAILPW00  BACKUP YOUR EXISTING POWER QUEUES
* ISSUE THE POWER POFFLOAD COMMAND
* THE FORMAT IS
* POFFLOAD BACKUP,ALL,CUU,DO
// PAUSE AFTER SUCCESSFUL POFFLOAD, PRESS 0 AND ENTER
/*
/&
* $$ EOJ
```



```

* $$ JOB JNM=TAILPW20,CLASS=0,DISP=L
// JOB TAILPW20  SAVE PROCEDURES FROM IJSYSRS.SYSLIB TO PRD2.TAILSAVE
// EXEC LIBR,PARM='MSHP'
ACC S=PRD2.TAILOR          Clean-up previously used auxiliary
DELETE *.PROC              sublibraries
ACC S=PRD2.TAILSAVE
DELETE *.PROC
CONNECT S=IJSYSRS.SYSLIB:PRD2.TAILSAVE
COPY *.PROC R=Y
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=TAILPW30,CLASS=0,DISP=L
// JOB TAILPW30  UPDATE DTRPOWR PROCEDURE
// EXEC LIBR,PARM='MSHP'
ACCESS S=PRD2.TAILOR
CATALOG DTRPOWR.PROC      DATA=YES REPLACE=YES
// ASSGN SYS000,DISK,VOL=SYSWK1,SHR      POWER ACCOUNT FILE
// ASSGN SYS001,DISK,VOL=DOSRES,SHR      POWER QUEUE FILE
// ASSGN SYS002,DISK,VOL=SYSWK1,SHR      POWER DATA FILE 1
// ASSGN SYS003,DISK,VOL=SYSWK2,SHR      POWER DATA FILE 2
/+
/&
* $$ EOJ

```

```

* $$ JOB JNM=TAILPW40,CLASS=0,DISP=L
// JOB TAILPW40 UPDATE STDLABEL PROCEDURE
// EXEC LIBR,PARM='MSHP'
ACCESS S=PRD2.TAILOR
CATALOG STDLABEL.PROC          DATA=YES REPLACE=YES
// OPTION STDLABEL
/. STANDARD LABELS FOR 3370
.
.
.
// DLBL IJQFILE,'VSE.POWER.QUEUE.FILE',1999/365,DA
// EXTENT SYS001,DOSRES,1,0,55862,1488
.
.
.
// DLBL DTSFILE,'ICCF.LIBRARY',1999/365,DA
// EXTENT SYS010,SYSWK1,1,0,327766,50000
// EXTENT SYS011,SYSWK2,1,1,254000,50000
// DLBL IJAFILE,'VSE.POWER.ACCOUNT.FILE',1999/365,DA
// EXTENT SYS000,SYSWK1,1,0,446016,2232
.
.
.
// DLBL IJDFILE,'VSE.POWER.DATA.FILE',1999/365,DA
// EXTENT SYS002,SYSWK1,1,0,394816,51200
// EXTENT SYS003,SYSWK2,1,1,304000,51200
// EXEC PROC=STDLABUP - INCLUDE VSAM FULIST FILE LABELS
// EXEC PROC=STDLABUS - INCLUDE USER STANDARD AND PARSTD LABELS
// OPTION USRLABEL - TO CLEAR LABEL AREA AND RESET LOCK
/*
/+
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=TAILPW50,CLASS=0,DISP=L
// JOB TAILPW50 CATALOG STARTUP-PROCEDURE FOR POWER PARTITION
// EXEC LIBR
ACC S=PRD2.TAILOR
CATALOG $1JCL370.PROC DATA=YES REPLACE=YES
// JOB PWRNORM - START POWER FOR NON-SHARED ENVIRONMENT
// EXEC PROC=DTRPOWR GET POWER FILES ASSGN'S
// EXEC PROC=DTRICCF GET DTSFILE ASSGN'S
// LIBDEF *,SEARCH=(PRD2.CONFIG,PRD1.BASE),PERM
// LIBDEF DUMP,CATALOG=SYSDUMP.F1,PERM
// EXEC PWRNORM
FORMAT=D,A
PSTART BG,A0I
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART F2,J2
READER=FEC
PRINTERS=FEE,FEF
PUNCHES=FED
PSTART F3,K3
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART F4,L4
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART F5,M5
READER=FEC
PRINTERS=FEE
PUNCHES=FED
/*
/+
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=TAILPW60,CLASS=0,DISP=L
// JOB TAILPW60 COPY PROCEDURES TO IJSYSRS.SYSLIB AND PRD2.SAVE
// EXEC LIBR,PARM='MSHP'
CONNECT S=PRD2.TAILOR:PRD2.SAVE
COPY *.PROC R=Y
CONNECT S=PRD2.TAILOR:IJSYSRS.SYSLIB
COPY *.PROC R=Y
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=TAILPW70,CLASS=0,DISP=L
// JOB TAILPW70 POWER SHUT-DOWN AND BRING-UP
* ISSUE THE FOLLOWING COMMAND SEQUENCE TO SHUT DOWN
* CICS, VTAM, AND POWER
* /TC
* 23 CEMT P SHUT IMM
* Z NET,QUICK
* PEND
*
* RE-IPL THE SYSTEM
* POFFLOAD LOAD,ALL,CUU
* R RDR,VTAMSTRT
* R RDR,CICSICCF
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=TAILPW80,CLASS=0,DISP=L
// JOB TAILPW80 RECATALOG STARTUP-PROCEDURE FOR POWER PARTITION
* WITHOUT FORMATTING THE POWER FILES
// EXEC LIBR
ACC S=IJSYSRS.SYSLIB
CATALOG $1JCL370.PROC DATA=YES REPLACE=YES
// JOB PWRNORM - START POWER FOR NON-SHARED ENVIRONMENT
// EXEC PROC=DTRPOWER GET POWER FILES ASSGN'S
// EXEC PROC=DTRICCF GET DTSFILE ASSGN'S
// LIBDEF *,SEARCH=(PRD2.CONFIG,PRD1.BASE),PERM
// LIBDEF DUMP,CATALOG=SYSDUMP.F1,PERM
// EXEC PWRNORM
FORMAT=NO
PSTART BG,AOI
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART F2,J2
READER=FEC
PRINTERS=FEE,FEF
PUNCHES=FED
PSTART F3,K3
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART F4,L4
READER=FEC
PRINTERS=FEE
PUNCHES=FED
PSTART F5,M5
READER=FEC
PRINTERS=FEE
PUNCHES=FED
/*
/+
CONNECT S=IJSYSRS.SYSLIB:PRD2.SAVE
COPY $1JCL370.PROC REPLACE=YES
/*
/&
* $$ EOJ

```

OTHER FILES

Summary of Jobs

- | | |
|----------|---|
| TAILOT00 | Save original procedures from IJSYSRS.SYSLIB into an auxiliary sublibrary.
Store new STDLABEL procedure into a different auxiliary sublibrary. |
| TAILOT10 | Relocate and copy System History File. |
| TAILOT20 | Relocate and copy Job Manager File. |
| TAILOT30 | Create new IPL procedure to reflect the new location of System History File, Recorder and Hardcopy File. |
| TAILOT40 | Activate the new STDLABEL procedure. |
| TAILOT50 | Unbatch all partitions except BG. |
| TAILOT60 | Delete the old DUMP Library. |
| TAILOT70 | Define the new extended DUMP Library. |
| TAILOT80 | Scratch the VTOC entries of the old files. |


```

* $$ JOB JNM=TAILOT00,CLASS=0,DISP=L
// JOB TAILOT00  SAVE ORIGINAL PROCS AND CATAL NEW STDLABEL
// EXEC LIBR,PARM='MSHP'
ACC S=PRD2.TAILOR
DELETE *.*
CONNECT S=IJSYSRS.SYSLIB:PRD2.TAILSAVE
COPY *.PROC R=Y
ACC S=PRD2.TAILOR
CATALOG STDLABEL.PROC          DATA=YES REPLACE=YES
// OPTION STDLABEL

.

// DLBL SYSDUMP,'VSE.DUMP.LIBRARY',1999/365,SD
// EXTENT ,SYSWK1,1,0,194742,37200
// EXTENT ,SYSWK2,1,1,6272,37200

.

// DLBL VSEJMGR,'VSESP.JOB.MANAGER.FILE',1999/365,SD
// EXTENT SYSREC,SYSWK2,1,0,2,126

.

// DLBL IJSYSHF,'VSE.SYSTEM.HISTORY.FILE',1999/365,SD
// EXTENT SYSREC,SYSWK2,1,0,2176,4096

.

// DLBL IJSYSCN,'VSE.HARDCOPY.FILE',1999/365,SD
// EXTENT SYSREC,SYSWK2,1,0,355200,4994

.

// DLBL IJSYSRC,'VSE.RECORDER.FILE',1999/365,SD
// EXTENT SYSREC,SYSWK2,1,0,360194,752

.

// EXEC PROC=STD LABUP - INCLUDE VSAM FULIST FILE LABELS
// EXEC PROC=STD LABUS - INCLUDE USER STANDARD AND PARSTD LABELS
// OPTION USRLABEL - TO CLEAR LABEL AREA AND RESET LOCK
/*
/+
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=TAILOT10,CLASS=0,DISP=L
// JOB TAILOT10   RELOCATE THE SYSTEM HISTORY FILE
// DLBL IJSYSHF,'VSE.SYSTEM.HISTORY.FILE',1999/365,SD
// EXTENT SYSREC,SYSWK1,1,0,476550,3460
// DLBL IJSYS02,'VSE.SYSTEM.HISTORY.FILE',1999/365,SD
// EXTENT SYS002,SYSWK2,1,0,2176,4096
// ASSGN SYS002,DISK,VOL=SYSWK2,SHR
// EXEC MSHP
COPY HIST SYS AUX
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=TAILOT20,CLASS=0,DISP=L
// JOB TAILOT20   RELOCATE THE JOB MANAGER FILE
// DLBL OLDJMGR,'VSESP.JOB.MANAGER.FILE'
// EXTENT SYS004,SYSWK1,1,0,470704,100
// DLBL VSEJMGR,'VSESP.JOB.MANAGER.FILE',1999/365,SD
// EXTENT SYS005,SYSWK2,1,0,2,126
// ASSGN SYS004,DISK,VOL=SYSWK1,SHR
// ASSGN SYS005,DISK,VOL=SYSWK2,SHR
// EXEC FCOPYFB
COPY INFILE=OLDJMGR OUTFILE=VSEJMGR -
REORGANIZE NOVERIFY LIST
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=TAILOT30,CLASS=0,DISP=L
// JOB TAILOT30      RE-ASSGN SYSREC IN $IPL370
* ****
* **
* ** THIS JOB IS JUST A REMINDER
* ** USE 'II' TO CHANGE THE
* ** ASSIGNMENT OF SYSREC IN $IPL370
* ** SYSREC SHOULD NOW POINT TO 'SYSWK2'
* **
* ****
/*
/&
* $$ EOJ

```

TAS\$ICM5 TAILOR IPL PROCEDURE: SYSREC/SYSCAT ASSIGNMENT

Enter the required data and press ENTER.

SYSREC..... SYSWK2 Volume Serial Number of DASD where
the recorder
resides on

SYSCAT..... DOSRES Volume Serial Number of DASD where
the VSAM master catalog resides on

PF1=HELP

3=END

5=RETRY

=>

```

* $$ JOB JNM=TAILOT40,CLASS=0,DISP=L
// JOB TAILOT40    COPY NEW PROCS FROM PRD2.TAILOR
// EXEC LIBR,PARM='MSHP'
CONNECT S=PRD2.TAILOR:IJSYSRS.SYSLIB
COPY *.PROC R=Y
CONNECT S=PRD2.TAILOR:PRD2.SAVE
COPY *.PROC R=Y
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=TAILOT50,CLASS=0,DISP=L
// JOB TAILOT50    UNBATCH ALL PARTITIONS EXCEPT BG
* ****
* **              ***
* **  T H I S   J O B   I S   J U S T   A   R E M I N D E R      ***
* **              ***
* **  I S S U E   T H E   F O L L O W I N G   C O M M A N D   S E Q U E N C E :      ***
* **              ***
* **      /TC      ***
* **      23 CEMT P SHUT IMM      ***
* **      23 EOB      ***
* **      Z NET,QUICK      ***
* **      PEND      ***
* **      1 /&      ***
* **      1 UNBATCH      ***
* **      2 UNBATCH      ***
* **      3 UNBATCH      ***
* **      4 UNBATCH      ***
* **      5 UNBATCH      ***
* **      0 ASSGN SYSIN,CUU      PHYSICAL OR VM RDR      ***
* **      0 ASSGN SYSLST,CUU      PHYSICAL OR VM PRT      ***
* **              ***
* **  E X E C U T E   T A I L O T 6 0 / 7 0   V I A   V M   S U B M I S S I O N   O R   S Y   S L O G      ***
* **              ***
* ****
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=TAILOT60,CLASS=0,DISP=L
// JOB TAILOT60      DELETE OLD DUMP LIBRARY
// PAUSE **** ENSURE NO DUMP FILES ARE ACTIVE ****
// LIBDROP *,PERM
// DLBL SYSDUMP,'VSE.DUMP.LIBRARY',1999/365,SD
// EXTENT ,SYSWK1,1,0,194742,37200
// EXEC LIBR,PARM='MSHP'
DELETE S=SYSDUMP.F1
DELETE S=SYSDUMP.F2
DELETE S=SYSDUMP.F3
DELETE S=SYSDUMP.F4
DELETE S=SYSDUMP.F5
DELETE S=SYSDUMP.F6
DELETE S=SYSDUMP.F7
DELETE S=SYSDUMP.F8
DELETE S=SYSDUMP.F9
DELETE S=SYSDUMP.FA
DELETE S=SYSDUMP.FB
DELETE S=SYSDUMP.BG
DELETE L=SYSDUMP
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=TAILOT70,CLASS=0,DISP=L
// JOB TAILOT70      CREATE NEW DUMP LIBRARY
// DLBL SYSDUMP, 'VSE.DUMP.LIBRARY',1999/365,SD
// EXTENT ,SYSWK1,1,0,194742,37200
// EXTENT ,SYSWK2,1,1,6272,37200
// EXEC LIBR,PARM='MSHP'
DEFINE L=SYSDUMP
DEFINE S=SYSDUMP.F1 REUSE=IMM
DEFINE S=SYSDUMP.F2 REUSE=IMM
DEFINE S=SYSDUMP.F3 REUSE=IMM
DEFINE S=SYSDUMP.F4 REUSE=IMM
DEFINE S=SYSDUMP.F5 REUSE=IMM
DEFINE S=SYSDUMP.F6 REUSE=IMM
DEFINE S=SYSDUMP.F7 REUSE=IMM
DEFINE S=SYSDUMP.F8 REUSE=IMM
DEFINE S=SYSDUMP.F9 REUSE=IMM
DEFINE S=SYSDUMP.FA REUSE=IMM
DEFINE S=SYSDUMP.FB REUSE=IMM
DEFINE S=SYSDUMP.BG REUSE=IMM
/*
* *** R E - I P L  AFTER SUCCESSFUL EXECUTION
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=TAILOT80,CLASS=0,DISP=L
// JOB TAILOT80      SCRATCH OLD FILES
* ****
* **
* **  START DITTO VIA THE FOLLOWING COMMAND SEQUENCE
* **
* **    R RDR,PAUSEBG
* **
* **    0 // EXEC DITTO
* **
* **    0 PVT,VSYSWK1,SCRATCH,VSESP.JOB.MANAGER.FILE
* **
* **    0 PVT,VSYSWK1,SCRATCH,VSE.SYSTEM.HISTORY.FILE
* **
* **    0 PVT,VSYSWK1,SCRATCH,VSE.HARDCOPY.FILE
* **
* **    0 PVT,VSYSWK1,SCRATCH,VSE.RECORDER.FILE
* **
* **    0 EOJ
* **
* **
* ****
/*
/&
* $$ EOJ

```

SIZING OPTIONAL PRODUCTS

From the IESEADM screen the INSTALLATION screen is requested by entering **1**.

IESADMSL.IESEADM

VSE/SP FUNCTION SELECTION

Enter the number of your selection and press the ENTER key:

- 1 Installation
- 2 Resource Definition
- 3 Operations
- 4 Problem Handling
- 5 Program Development
- 6 Command Mode

PF1=HELP

3=SIGN OFF
9=Escape(m)

6=ESCAPE(U)

==> 1

When in the INSTALLATION screen, enter **1** to go to the Install Programs - (V2 Format) screen.

IESADMSL.IESEINST		INSTALLATION	
Enter the number of your selection and press the ENTER key:			
<u>1</u> Install Programs - (V2 Format)			
2 Install Programs - (V1 Format)			
3 Install Generation Feature			
4 IBM Service			
5 Install Network Tape			
PF1=HELP	3=END	4=RETURN	6=ESCAPE(U)
	9=Escape(m)		
==> <u>1</u>		Path: 1	

You now get a screen with 3 choices. Select item **1**. This scans the optional products tape and creates a table with the number of library blocks required for each product.

IESADMSL.IESEPROV

INSTALL PROGRAMS - (V2 FORMAT)

Enter the number of your selection and press the ENTER key:

- 1 Prepare for Installation
- 2 Install Product(s) from Tape
- 3 Complete Installation

PF1=HELP

3=END

4=RETURN

6=ESCAPE(U)

9=Escape(m)

==> 1

Path: 11

On the next screen, you can enter the address of the **tape drive**.

```
INS$PRI2          PREPARE INSTALL PROGRAMS (V2 FORMAT)
Enter the required data and press ENTER.

TAPE ADDRESS.....1B3          Address of input tape unit (cuu)

PF1=HELP          3=END          5=RETRY

==>
```

The next screen allows you to enter **batch job disposition parameters**. The suggested values are listed below:

SUB\$PRO5		JOB DISPOSITION	
Enter the required data and press ENTER.			
JOB DESTINATION.....	<u>3</u>	Enter 1 to submit the job to batch. Enter 2 to file in library Enter 3 to do both.	
JOB NAME.....	INSPRE	Enter the name under which the job will be saved in VSE/ICCF.	
PRIORITY.....	<u>3</u>	Priority 0-9 for this job.	
CLASS.....	<u>0</u>	An asterisk may not be changed	
DISPOSITION.....	<u>D</u>	Enter disposition as D,H,K, or L. An An asterisk may not be changed.	
JOB ACCOUNTING.....			
HOLD LIST IN QUEUE.....	<u>2</u>	Enter 1 to hold output in list queue Enter 2 to print output immediately.	
OTHER PARAMETERS.....	<u>2</u>	Enter 1 if you would like to change any of the other POWER JOB para- meters. Otherwise enter 2.	
PF1=HELP		3=END	5=RETRY
==>			

After the job has run, you can get a **list of the products** on the tape and you can verify the **total number of library blocks** required for those products. To do this, enter **2** on the screen below.

IESADMSL.IESEPROV

INSTALL PROGRAMS - (V2 FORMAT)

Enter the number of your selection and press the ENTER key:

- 1 Prepare for Installation
- 2 Install Product(s) from Tape
- 3 Complete Installation

PF1=HELP

3=END

4=RETURN

6=ESCAPE(U)

9=Escape(m)

==> 2

Path: 11

The resultant screen (shown below) shows both which products will be installed if the ENTER key is pressed and that the number of library blocks necessary for all the products to be installed is **113214** blocks.

```

INS$OPI1                      INSTALL PROGRAMS (V2 FORMAT)

LIST OF PROGRAMS TO BE INSTALLED

OPTIONS:  1 = INSTALL      2 = SKIP INSTALLATION

      OPT      IDENTIFIER      LIBRARY  SUBLIBR.  SEQ.NO.  TAPE NO.
      NAME      NAME
      1      SORT/MERGE..2.5.0 PRD2      PROD      1      1
      1      COBOL.....1.3.0 PRD2      PROD      2      1
      1      PL/I OPT...1.6.0 PRD2      PROD      3      1
      1      PL/I TRANS..1.6.0 PRD2      PROD      4      1
      1      PL/I RES...1.6.0 PRD2      PROD      5      1
      1      PL/I COMP..1.6.0 PRD2      PROD      6      1
      1      RPGII.....1.3.0 PRD2      PROD      7      1
      1      SQL/DS.....1.2.0 PRD2      DBASE      8      1
      1      DATA DICT..1.4.0 PRD2      DBASE      9      1

PF1=HELP                      3=END                      5=RETRY      6=PROCESS
                        8=FORWARD
ALL SCANNED PRODUCTS NEED 113214 LIBRARY BLOCKS ON DISK.
==>

```

Please note that whereas the above listed products represent only those first found on the install tape, the number of library blocks is for **all** products on the tape which are to be installed.

SETSDL PROCEDURE

```
* $$ JOB JNM=SETSDL,CLASS=0,DISP=D,NTFY=YES
* $$ LST DISP=D,CLASS=R,DEST=(*,MERZ)
// JOB SETSDL CATALOG SETSDL PROCEDURE
// EXEC LIBR,PARM='MSHP'
ACCESS S=IJSYSRS.SYSLIB
CATALOG SETSDL.PROC DATA=YES REPLACE=YES
SET SDL
  $$BOCRTA,MOVE DOC TRANSIENT
  $$BOCRTC,MOVE DOC TRANSIENT
  $$BOCRTD,MOVE DOC TRANSIENT
  $$BOCRTG,MOVE DOC TRANSIENT
  $$BOCRTH,MOVE DOC TRANSIENT
  $$BOCRTQ,MOVE DOC TRANSIENT
  $$BOCRTZ,MOVE DOC TRANSIENT
  $$BATTNA,MOVE ATTN ROUTINE
  $$BATTNB,MOVE ATTN ROUTINE
  $$BATTNH,MOVE ATTN ROUTINE
  $$BOESTV,MOVE TAPE STATISTICS
  $$BOKUL1,MOVE CKD LABELS
  $$BOMLTA,MOVE SVMSG RETURN
  $$BOMSVA,MOVE SVMSG CALL
  $$BOMSV2,MOVE SVMSG CALL
  $$BOPEN,MOVE OPEN TRANSIENT
  $$BOPEN1,MOVE OPEN TRANSIENT
  $$BOPEN2,MOVE OPEN TRANSIENT
  $$BOIGN,MOVE OPEN TRANSIENT
  $$BOPLBL,MOVE LABEL TRANSIENT
  $$BCLOSE,MOVE CLOSE TRANSIENT
  $$BCLOS2,MOVE CLOSE TRANSIENT
  $$BCLOS5,MOVE CLOSE TRANSIENT
  $$BCLRPS,MOVE CLOSE RPS TRANSIENT
  $$BOPENR,MOVE OPENR TRANSIENT
  $$BOPNR2,MOVE OPENR TRANSIENT
  $$BOPNR3,MOVE OPENR TRANSIENT
  $$BOIS03,MOVE OPEN ERRMSG
  $$BOUR01,MOVE OPEN TRANSIENT
  $$BOSDC1,MOVE CLOSE TRANSIENT
  $$BOCP01,MOVE OPEN TRANSIENT
  $$BOCP03,MOVE OPEN TRANSIENT
```

SETSDL PROCEDURE (cont'd)

```
    $$BCVS02,MOVE    VSAM
    $$BCVSAM,MOVE    VSAM
    $$BOVSAM,MOVE    VSAM
    $$BSETL,MOVE     ISAM
    $$BSETL1,MOVE    ISAM
    DTSCDUMP,SVA     ICCF
    DTSPROCS,SVA     ICCF
    DTSSBMT1,SVA     ICCF
    LINKNGO2,SVA     ICCF
    DTSXTRCT,SVA     ICCF
    DTSIPWRP,SVA     ICCF
/*
/+
/*
/&
* $$ EOJ
```

Note: This procedure is identical with the one delivered with the VSE/SP system.

CHAPTER 4. JOB CONTROL ANALYSIS AND CONVERSION

HIGHLIGHTS

This chapter covers the following topics:

- Samples of how to use the **Job Control Scanner** program
- A sample of a job control exit routine which shows how to prevent the unauthorized usage of **PARM=MSHP** for **LIBR** and **LNKEDT**
- ICCF procedures and jobs
 - A modified **SUBMIT** procedure which allows the automatic setting of the **POWER DEST** parameter
 - **BACKUP/RESTORE** samples for the ICCF **DTSFILE**
- A lable procedure skeleton as provided in **ICCF library 59**
- An ASI master procedure (**\$ASIPROC**) as established for a multi-CPU /multi-VSE guest machine environment
- How to code an **ASSEMBLER** subroutine which enables **HLL** programs to make use of conditional job control
- Virtual I/O
 - An explanation of the principles of VIO
 - Sample ICCF **CLISTs** and **CMS EXECs** which are using **// OPTION CATAL** for **LINK** and **GO** (with an automatic deletion of the library member at the end of the execution step).

JOB CONTROL SCANNER

Input from ICCF Library

Sample Jobs

```
* $$ JOB JNM=EX41SCAN,CLASS=0,DISP=D
// JOB EX41SCAN PUNCH ICCF MEMBERS TO TAPE FOR PROCESSING
*          BY JOB CONTROL SCANNER
// ASSGN SYS010,DISK,VOL=SYSWK1,SHR    DTSFILE EXTENT #1
// ASSGN SYS011,DISK,VOL=SYSWK2,SHR    DTSFILE EXTENT #2
// ASSGN SYSPCH,1B1,D0                OUTPUT TAPE 6250 BPI
// MTC REW,SYSPCH
// EXEC DTSUTIL
PUNCH LIBRARY(98) PUNCTL
/*
// MTC WTM,SYSPCH,2
// MTC REW,SYSPCH
/*
/&
* $$ EOJ
```

```
* $$ JOB JNM=EX42SCAN,CLASS=0,DISP=D
// JOB EX42SCAN PROCESS ICCF MEMBERS BY JOB CONTROL SCANNER
// ASSGN SYS007,1B1          INPUT TAPE (CREATED BY DTSUTIL)
// MTC REW,SYS007
// EXEC JCLSCAN
I
/*
// MTC RUN,SYS007
/&
* $$ EOJ
```

Sample Output

S C A N - P R O G R A M F O R M I G R A T I O N

```
LIBRARY 98      MEMBER IND098
NO INCOMPATIBLE STATEMENTS
LIBRARY 98      MEMBER JCSJOBA
NO INCOMPATIBLE STATEMENTS
LIBRARY 98      MEMBER P50L412
NO INCOMPATIBLE STATEMENTS
LIBRARY 98      MEMBER PF
NO INCOMPATIBLE STATEMENTS
LIBRARY 98      MEMBER S31L501
// LIBDEF SL, FROM=S31USL2
*** STATEMENT WILL BE MIGRATED BY THE SYSTEM, BUT SHOULD BE MODIFIED ***
// EXEC ESERV
*** PLEASE CHECK SYSIPT DATA ****
END OF MEMBER S31L501
LIBRARY 98      MEMBER JCSJOB2
NO INCOMPATIBLE STATEMENTS
LIBRARY 98      MEMBER RESTTEST
NO INCOMPATIBLE STATEMENTS
LIBRARY 98      MEMBER P50L411
NO INCOMPATIBLE STATEMENTS
LIBRARY 98      MEMBER CAPPUTIL
NO INCOMPATIBLE STATEMENTS
LIBRARY 98      MEMBER JCSJOB3
NO INCOMPATIBLE STATEMENTS
LIBRARY 98      MEMBER JCSJOB1
NO INCOMPATIBLE STATEMENTS
LIBRARY 98      MEMBER NED
NO INCOMPATIBLE STATEMENTS
LIBRARY 98      MEMBER AMSLCAT
NO INCOMPATIBLE STATEMENTS
LIBRARY 98      MEMBER JCSJOBB
NO INCOMPATIBLE STATEMENTS
```

E N D O F S C A N P R O G R A M

Input from PL / SSL

Sample Jobs

```
* $$ JOB JNM=EX44SCAN,CLASS=0,DISP=D
// JOB EX44SCAN          PUNCH MEMBERS FROM PROCEDURE LIBRARY TO TAPE
*                        PROCESSING BY JOB CONTROL SCANNER
// ASSGN SYSPCH,1B1,DO   OUTPUT TAPE 6250 BPI
// MTC REW,SYSPCH
// EXEC PSERV
PUNCH ALL
/*
// MTC WTM,SYSPCH,2
// MTC REW,SYSPCH
/&
* $$ EOJ
```

```
* $$ JOB JNM=EX45SCAN,CLASS=0,DISP=D
// JOB EX45SCAN          PUNCH MEMBERS FROM SSL (POWER SLI BOOKS) TO
*                        TAPE FOR PROCESSING BY JOB CONTROL SCANNER
// ASSGN SYSPCH,1B1,DO   OUTPUT TAPE 6250 BPI
// MTC REW,SYSPCH
// EXEC SSERV
PUNCH P.ALL
/*
// MTC WTM,SYSPCH,2
// MTC REW,SYSPCH
/&
* $$ EOJ
```

```

* $$ JOB JNM=EX46SCAN,CLASS=0,DISP=D
// JOB EX46SCAN      PROCESS DOS/VSE LIBRARY MEMBERS BY
*                   JOB CONTROL SCANNER
// ASSGN SYS007,1B1  INPUT TAPE (CREATED BY PSERV / SSERV)
// ASSGN SYSPCH,1B2  OUTPUT TAPE (MEMBERS IN ERROR)
// MTC REW,SYS007
// MTC REW,SYSPCH
// EXEC JCLSCAN
D,MERZ,21           MERZ=ICCF USER-ID. 21=TARGET ICCF LIBRARY
/*
// MTC RUN,SYS007
// MTC WTM,SYSPCH
// MTC REW,SYSPCH
/&
* $$ EOJ

```

```

* $$ JOB JNM=EX48SCAN,CLASS=0,DISP=D
// JOB EX48SCAN      RESTORE ERRANEOUS PL/SSL MEMBERS TO ICCF LIBRARY
                        FOR FURTHER PROCESSING
// ASSGN SYS010,DISK,VOL=SYSWK1,SHR  DTSFILE EXTENT #1
// ASSGN SYS011,DISK,VOL=SYSWK2,SHR  DTSFILE EXTENT #2
// ASSGN SYSIPT,1B2    TAPE (CREATED BY JCL SCANNER)
// MTC REW,SYSIPT
// EXEC DTSUTIL
/*
// MTC RUN,SYSIPT
/*
/&
* $$ EOJ

```

Sample Output

```
      S C A N   -   P R O G R A M       F O R       M I G R A T I O N

LIBRARY 20      MEMBER DELECDKC
// EXEC MAINT
*** PLEASE CHECK SYSIPT DATA ****
END OF MEMBER DELECDKC

*** FOLLOWING MEMBER DOES NOT CONTAIN INCOMPATIBLE STATEMENTS: ****
      CATALP  FASTFTCH,VM=3.5
LIBRARY 20      MEMBER DELELNKP

LIBRARY 20      MEMBER $1JCLVSP
LIBDEF SL,SEARCH=PRDSLA
*** STATEMENT WILL BE MIGRATED BY THE SYSTEM,BUT SHOULD BE MODIFIED ***
END OF MEMBER $1JCLVSP

*** FOLLOWING MEMBER DOES NOT CONTAIN INCOMPATIBLE STATEMENTS: ****
      CATALP  $2JCLVSP,VM=0.0
LIBRARY 20      MEMBER DLIPROC
// LIBDEF CL,FROM=PRDCLF,SEARCH=(PRDCLF,USRCL1,PRDCLG)
*** STATEMENT WILL BE MIGRATED BY THE SYSTEM,BUT SHOULD BE MODIFIED ***
END OF MEMBER DLIPROC

*** FOLLOWING MEMBER DOES NOT CONTAIN INCOMPATIBLE STATEMENTS: ****
      CATALP  $OPARSTD,VM=0.0

      E N D       O F       S C A N       P R O G R A M
```

Input from CMS

Sample EXEC / Job

```
&TRACE
*
* *****
*
*   SAMPLE CMS EXEC PROCEDURE
*
*   TRANSFERS CMS FILES TO TAPE FOR SCANNING
*
*   OUTPUT TAPE HAS TO BE ATTACHED TO 181
*
* *****
*
CH RDR CL J HOLD
CP SP PUN * CL J
*
* *****
*
*   PUNCH ALL FILES CONTAINING DOS/VSE JCL
*
*   FORMAT IS:
*
*   PUNCH FN FT FM
*
* *****
*
CP SP RDR CL J CONT
FI INMOVE READER
FI OUTMOVE TAP1 (LRECL 80 BLOCK 80 RECFM F DEN 6250
MOVEFILE
TAPE WTM
TAPE REW
CP SP PUN OFF CL A
CP SP RDR NOCONT CL A
```

Note: For a more sophisticated CMS EXEC, please refer to "CMS EXEC Samples" on page 345.

```
* $$ JOB JNM=EX410SCN,CLASS=0,DISP=D
// JOB EX410SCN  PROCESS CMS FILES BY JOB CONTROL SCANNER
// ASSGN SYS007,1B1      INPUT TAPE (FROM CMS EXEC)
// MTC REW,SYS007
// EXEC JCLSCAN
C
/*
// MTC RUN,SYS007
/&
* $$ EOJ
```


Sample Output

```
          S C A N   -   P R O G R A M       F O R       M I G R A T I O N

CMS FILE :  Q1JOB      JOB      A1
// EXEC CORGZ
*** PLEASE CHECK SYSIPT DATA ****
      END OF : Q1JOBJOB      A1

CMS FILE :  Q2JOB      JOB      A1
// LIBDEF CL,NEW=S31UCL2
*** STATEMENT WILL BE MIGRATED BY THE SYSTEM,BUT SHOULD BE MODIFIED ***
// LIBDEF RL,NEW=S31URL2
*** STATEMENT WILL BE MIGRATED BY THE SYSTEM,BUT SHOULD BE MODIFIED ***
// LIBDEF SL,NEW=S31USL2
*** STATEMENT WILL BE MIGRATED BY THE SYSTEM,BUT SHOULD BE MODIFIED ***
// LIBDEF PL,NEW=S31UPL2
*** STATEMENT WILL BE MIGRATED BY THE SYSTEM,BUT SHOULD BE MODIFIED ***
// EXEC CORGZ
*** PLEASE CHECK SYSIPT DATA ****
      END OF : Q2JOBJOB      A1

CMS FILE :  Q9JOB      JOB      A1
// LIBDEF CL,FROM=S31UCL1
*** STATEMENT WILL BE MIGRATED BY THE SYSTEM,BUT SHOULD BE MODIFIED ***
// EXEC CORGZ
*** PLEASE CHECK SYSIPT DATA ****
      END OF : Q9JOBJOB      A1

          E N D       O F       S C A N       P R O G R A M
```

Input from Unstructured Files

Sample Job

```
* $$ JOB JNM=EX412SCN,CLASS=0,DISP=D
// JOB EX412SCN  PROCESS UNSTRUCTURED FILES BY JOB CONTROL SCANNER
// ASSGN SYS007,1B1      INPUT TAPE (FOR EXAMPLE, FROM CARD INPUT)
// MTC REW,SYS007
// EXEC JCLSCAN
U
/*
// MTC RUN,SYS007
/&
* $$ EOJ
```

Sample Output

```
          S C A N   -   P R O G R A M       F O R       M I G R A T I O N
          L I S T I N G   O F   S C A N N E D   S T A T E M E N T S

* $$ JOB JNM=S31L101,CLASS=0,DISP=D
// JOB S31L101  CREATE NEW SYSTEM RESIDENCE
* SIPO 3.1 LIBRARIAN  ID=1  JOB=01
// DLBL IJSYSRS,'NEW.SYSRES.S31',99/365,SD
// EXTENT SYS002,SIPO31,1,0,2,35838
// ASSGN SYS002,DISK,VOL=SIPO31,SHR
// EXEC CORGZ
*** PLEASE CHECK SYSIPT DATA ****
  ALLOC CL=33662(128),RL=0(0),SL=0(0),PL=1920(64)
/*
/&
* $$ EOJ

* $$ JOB JNM=S31L102,CLASS=0,DISP=D
// JOB S31L102  CREATE PRIVATE LIBRARIES
* SIPO 3.1 LIBRARIAN  ID=1  JOB=02
// LIBDEF CL,NEW=S31UCL2
*** STATEMENT WILL BE MIGRATED BY THE SYSTEM,BUT SHOULD BE MODIFIED ***
// LIBDEF RL,NEW=S31URL2
*** STATEMENT WILL BE MIGRATED BY THE SYSTEM,BUT SHOULD BE MODIFIED ***
// LIBDEF SL,NEW=S31USL2
*** STATEMENT WILL BE MIGRATED BY THE SYSTEM,BUT SHOULD BE MODIFIED ***
// LIBDEF PL,NEW=S31UPL2
*** STATEMENT WILL BE MIGRATED BY THE SYSTEM,BUT SHOULD BE MODIFIED ***
// EXEC CORGZ
*** PLEASE CHECK SYSIPT DATA ****
  NEWVOL CL=11200(128),RL=4032(64),SL=2816(64),PL=2432(64)
/*
/&
* $$ EOJ

          E N D       O F       L I S T I N G
          E N D       O F       S C A N       P R O G R A M
```

JOB CONTROL EXIT

* \$\$ JOB JNM=JOBEXIT,CLASS=0,DISP=D

* \$\$ LST CLASS=R,DISP=D,DEST=(*,IANH)

// JOB JOBEXIT PHASE

// LIBDEF *,CATALOG=IJSYSRS.SYSLIB

// OPTION CATAL,NODECK

PHASE \$JOBEXIT,S,NOAUTO,SVA,PBDY

// EXEC ASSEMBLY

*

	TITLE	'JOB CONTROL EXIT TO CHECK PARM=MSHP'
JOBEXIT	CSECT	
R0	EQU	0 SYSTEM ID 'SDOS'
R1	EQU	1 PARTITION COMREG ADDRESS
R2	EQU	2 SYSCOM ADDRESS
R3	EQU	3 JCL VECTOR TABLE ADDRESS
R4	EQU	4 JCL STATEMENT(S) BUFFER ADDRESS
R5	EQU	5 NUMBER OF CONTINUATION CARDS
R6	EQU	6
R7	EQU	7
R8	EQU	8
R9	EQU	9
R10	EQU	10
R11	EQU	11
R12	EQU	12
R13	EQU	13
R14	EQU	14 RETURN ADDRESS TO JOB CONTROL
R15	EQU	15 RETURN CODE FOR JOB CONTROL
	BALR	R11,0
	USING	*,R11 INITIALIZE BASE REGISTER

*

* THE FOLLOWING CODE CHECKS FOR UNAUTHORIZED USE OF THE 'MSHP'

* PARAMETER ON THE EXEC LIBR STATEMENT.

* THE USER IS AUTHORIZED IF HE HAS THE WORD 'MERZ' CODED AFTER

* THE EXEC STATEMENT.

*

	CLC	EXECRD,0(R3)	IS IT AN EXEC CARD?
	BNE	RETURN	NO ..NOT INTERESTED
	SR	R7,R7	CLEAR INDEXING REGISTER
	LA	R8,1	LOAD INCREMENT VALUE
	LA	R9,63	LOAD STATEMENT LENGTH (9-71)
LOOPLBR	LA	R10,8(R7,R4)	POINT TO INDEXED SPOT IN BUFFER
	CLC	LIBR,0(R10)	CHECK IF LIBR BEING EXECUTED
	BE	FOUNDLBR	YES....DO MORE CHECKING
	BXLE	R7,R8,LOOPLBR	CHECK REST OF CARD
	B	RETURN	NO LIBR...RETURN TO JOB CONTROL
FOUNDLBR	LR	R6,R10	SAVE START ADDRESS OF LIBR.
	LA	R10,5(R10)	INCREMENT TO START OF PARM FLD.
	CLC	PARM,0(R10)	TEST IF PARM IS CODED
	BNE	RETURN	NO PARM FIELD..LET IT GO
	LA	R10,6(R10)	INCREMENT TO PARM VALUE
	CLC	MSHP,0(R10)	TEST IF PARM = MSHP
	BE	GOTMSHP	YES..CHECK IF AUTHORIZED

	CLC	AMPS,0(R10)	TEST IF PARM IS A PARAMETER
*			IF SO, ASSUME THE WORSE!!.
	BNE	RETURN	NO ... LET IT GO.
GOTMSHP	LA	R9,56	LOAD STATEMENT LENGTH (25-80)
LOOPUSR	LA	R10,25(R7,R4)	POINT TO INDEXED SPOT.
	CLC	MERZ,0(R10)	IS USER AUTHORIZED??
	BE	ITSOK	YES ...LET IT GO.
	BXLE	R7,R8,LOOPUSR	CHECK REST OF CARD
	MVC	0(L'MSG,R6),MSG	REPLACE LIBR WITH XXXX ..GIVE MSG
	B	RETURN	RETURN TO JOB CONTROL
ITSOK	MVC	0(4,R10),BLANK	BLANK OUT AUTHORIZED FIELD
	B	RETURN	
*			
* END OF THE CODE TO CHECK FOR UNAUTHORIZED USE OF MSHP			
* PARAMETER ON THE EXEC LIBR STATEMENT.			
*			
RETURN	SR	R15,R15	RETURN CODE OK
	BR	R14	RETURN TO JOB CONTROL
EXECRD	DC	C'EXEC'	
LIBR	DC	C'LIBR'	
PARM	DC	C'PARM'	
MSHP	DC	C'MSHP'	
MERZ	DC	C'MERZ'	
BLANK	DC	C' '	
AMPS	DC	X'50'	
MSG	DC	C'XXXX ** UNAUTHORIZED USE OF -EXEC LIBR,PARM=MSHP- **'	
	END		
/*			
<u>// EXEC LNKEDT,PARM='MSHP'</u>			
/*			
<u>// EXEC LIBR,PARM='MSHP'</u>			
<u>CONN S=IJSYSRS.SYSLIB:PRD2.SAVE</u>			
<u>COPY \$JOBEXIT.PHASE R=Y</u>			
/*			
/&			
* \$\$ EOJ			

ICCF PROCEDURES / JOBS

SUBMIT Procedure Sample

The following SUBMIT procedure was modified to permit - by providing a new parameter (**DST=XXXX**) - the automatic creation of the **DEST** parameter in the POWER * \$\$ LST and * \$\$ PUN cards.

Modifications are underscored and marked with an 'M' in column 70.

```

* ****
*
* PROCEDURE TO SUBMIT A MEMBER DIRECTLY TO POWER/VS
* SUBMIT NNNN (PASS1) (DIRECT(BG)/RETURN(BG)) (PRINT) (PWD=PASS2)
*      (ICCFSLI) (DST=XXXX)
*
* ****
&&OPTIONS 0010011
&&IF &&PARMCT NE 0 &&GOTO START
&&TYPE ENTER  NAME (PASS1) (DIRECT(BG)/RETURN(BG)) (PRINT) (PWD=PASS2)
&&TYPE      (ICCFSLI) (DST=XXXX)
&&READ &&PARAMS
&&IF &&PARMCT EQ 0 &&EXIT
&&LABEL START
&&SET &&VARBLA  ' '          SET FOR NO DST= PARM
&&SET &&VARBL9  *            SET FOR NO DST= PARM
&&SET &&VARBLB  ' '          SET FOR NO DST= PARM
&&SET &&VARBL5  ' '          SET DEFAULT 'NO SLI'
&&SET &&VARBL6  &&PARAM1      SET MEMBER NAME
&&SHIFT 1                  SHIFT OUT MEMBER NAME
&&IF &&PARMCT EQ 0 &&GOTO NOPASS  MEMBER NAME ONLY SPECIF.
&&SET &&COUNT1 &&PARMCT
&&LABEL DSTLOOP              TEST IF 'DST=' SUPPLIED
&&SET &&VARBL8 &&SUBSTR 1 4 &&PARAM&&COUNT1
&&IF &&VARBL8 EQ DST= &&GOTO GOTDST  FOUND A 'DST=' PARAMETER
&&SET &&COUNT1 *-1
&&IF &&COUNT1 EQ 0 &&GOTO ENDSTL    ALL PARAMETERS SCANNED?
&&GOTO -DSTLOOP              TRY AGAIN
&&LABEL GOTDST
&&SET &&VARBL9 &&SUBSTR 5 8 &&PARAM&&COUNT1  SET USER NAME
&&SET &&VARBLA  '(,'
&&SET &&VARBLB  ')'
&&LABEL ENDSTL

```

```

&&SET &&COUNT1 &&PARMCT
&&LABEL SLILOOP
&&IF &&PARAM&&COUNT1 EQ ICCFSLI &&GOTO SLI
&&IF &&PARAM&&COUNT1 EQ NICCFSLI &&GOTO 6
&&SET &&COUNT1 *-1
&&IF &&COUNT1 EQ 0 &&GOTO ENDLOOP      ALL PARAMETERS SCANNED?
&&GOTO -SLILOOP
&&LABEL SLI
&&SET &&VARBL5 'ICCFSLI'                SET 'ICCFSLI'
&&SHIFT &&COUNT1                      SHIFT OUT ICCFSLI
&&LABEL ENDLOOP
&&SET &&COUNT1 &&PARMCT
&&LABEL PRTL1
&&IF &&COUNT1 LT 1 &&GOTO PRTEEND
&&IF &&PARAM&&COUNT1 EQ PRINT &&GOTO PRTL2
&&SET &&COUNT1 *-1
&&IF &&COUNT1 LT 1 &&GOTO PRTEEND
&&GOTO -PRTL1
&&LABEL PRTL2
&&IF &&COUNT1 EQ 0 &&GOTO PRTEEND
&&SET &&VARBL1 01                      SET UPSI FOR PRINT
&&SHIFT &&COUNT1                      SHIFT OUT PRINT
&&LABEL PRTEEND
&&SET &&VARBL3 '      '
&&SET &&COUNT1 &&PARMCT
&&SET &&COUNT1 *+1
&&LABEL PASSX
&&SET &&COUNT1 *-1
&&IF &&COUNT1 EQ 0 &&GOTO PASSY
&&SET &&VARBL4 &&SUBSTR 1 4 &&PARAM&&COUNT1
&&IF &&VARBL4 NE PWD= &&GOTO -PASSX
&&SET &&VARBL3 &&SUBSTR 5 8 &&PARAM&&COUNT1
&&SHIFT &&COUNT1                      SHIFT OUT PWD=PASSWORD
&&LABEL PASSY
&&IF &&VARBL6 NE ' ' &&GOTO PASS1
&&TYPE 'SUBMIT' IGNORED - NO MEMBER NAME SUPPLIED
&&EXIT
&&LABEL PASS1
&&IF &&PARMCT LT 1 &&GOTO NOPASS
&&LABEL NOPSS
&&SET &&VARBL2 &&SUBSTR 1 4 &&PARAM1
&&IF &&VARBL2 EQ DIRE &&GOTO NOPASS
&&IF &&VARBL2 EQ RETU &&GOTO NOPASS
&&IF &&VARBL2 EQ ICCF &&GOTO NOPASS
&&IF &&VARBL2 EQ NICC &&GOTO NOPASS
&&IF &&VARBL2 EQ PRIN &&GOTO NOPASS
&&IF &&VARBL2 EQ DST= &&GOTO NOPASS
&&IF &&PARAM1 EQ SEGMENT &&GOTO SEGM
&&SET &&VARBL0 &&VARBL2
&&SHIFT 1                            SHIFT OUT PASSWORD
&&LABEL NOPASS
/ LIST 1 1 &&VARBL6 &&VARBL0
&&IF &&RETCOD NE *FILE &&GOTO SOUR
&&TYPE *MEMBER &&PARAM1 NOT IN LIBRARY OR EMPTY

```

M

M

```

&&EXIT
&&LABEL SEGM
&&TYPE 'SUBMIT' IGNORED - SEGMENT-OPTION NO LONGER SUPPORTED
&&EXIT
&&LABEL SOUR
&&IF &&RETCOD NE *INVALID &&GOTO YESOR
&&TYPE *INVALID PASSWORD OR INVALID MEMBER NAME' &&PARAM1
&&EXIT
&&LABEL YESOR
&&IF &&RETCOD NE *MISSING &&GOTO OKSOR
&&TYPE *ENTER PASSWORD FOR MEMBER &&PARAM1
&&READ &&VARBL0
&&IF &&VARBL0 NE ' ' &&GOTO -NOPSS
&&EXIT
&&LABEL OKSOR
&&IF &&PARAM1 EQ ' ' &&GOTO LOAD
&&IF &&PARAM1 EQ ICCFSLI &&GOTO LOAD
&&IF &&PARAM1 EQ DIRECT &&GOTO LOAD
&&IF &&PARAM1 EQ PRINT &&GOTO LOAD
&&SET &&VARBL8 &&SUBSTR 1 4 &&PARAM1      SETUP FOR 'DST=' CHECKING      M
&&IF &&VARBL8 EQ DST= &&GOTO LOAD          CHECK FOR 'DST=' PARAM      M
&&SET &&VARBL7 &&PARAM1
&&SHIFT 1                                SHIFT OUT RETURN
&&LABEL LOAD
/INP NOPROMPT
&/LOAD DTSSUBMT
&&IF &&VARBL1 EQ 01 &&GOTO NOCONT
&/OPT JSDATA CONTINUE
&&GOTO CONT
&&LABEL NOCONT
&/OPT JSDATA
&&LABEL CONT
&/UPSI &&VARBL1
/PARM &&VARBL6 &&VARBL7 DIRECT P.&&VARBL3
* $$ JOB JNM=DIRECT,DISP=D,CLASS=A,NTFY=YES,
* $$      XDEST=***,LDEST=***,PDEST=***
* $$ LST DISP=D,CLASS=A,DEST=&&VARBLA&&VARBL9&&VARBLB      M
* $$ PUN DISP=(D),RBM=(1000),CLASS=(A),DEST=&&VARBLA&&VARBL9&&VARBLB      M
* $$ EOJ
* $$ JOB JNM=DIRECTBG,DISP=D,CLASS=A,NTFY=YES,      C
* $$      XDEST=***,LDEST=***,PDEST=***
* $$ LST DISP=D,CLASS=A,DEST=&&VARBLA&&VARBL9&&VARBLB      M
* $$ PUN DISP=(D),RBM=(1000),CLASS=(A),DEST=&&VARBLA&&VARBL9&&VARBLB      M
* $$ EOJ
* $$ JOB JNM=RETURN,DISP=D,CLASS=A,NTFY=YES,      C
* $$      XDEST=***,LDEST=***,PDEST=***
* $$ LST DISP=K,CLASS=Q,RBS=(0),DEST=***
* $$ PUN DISP=K,CLASS=Q,RBS=(0),DEST=***
* $$ EOJ
* $$ JOB JNM=RETURNBG,DISP=D,CLASS=A,NTFY=YES,      C
* $$      XDEST=***,LDEST=***,PDEST=***
* $$ LST DISP=K,CLASS=Q,RBS=(0),DEST=***
* $$ PUN DISP=K,CLASS=Q,RBS=(0),DEST=***
* $$ EOJ

```



```
XXXXXXXXXXXXXXXXX
&/INCLUDE &&VARBL6 &&VARBL0 &&VARBL5
&&LABEL NONAME
&&IF &&RETCOD NE *LIBRARY &&GOTO INPOKK
&&TYPE *LIBRARY OR LIBRARY DIRECTORY FULL
&&EXIT
&&LABEL INPOKK
/END
/PEND
/RUN
```

BACKUP/RESTORE of DTSFILE

```
* $$ JOB JNM=EX419BIF,CLASS=0,DISP=D
// JOB EX419BIF  BACKUP ICCF LIBRARY
// TLBL DTSBKUP,'ICCF.BACKUP'
// ASSGN SYS005,1B0,D0                OUTPUT TAPE 6250 BPI
// MTC REW,SYS005
// ASSGN SYS010,DISK,VOL=SYSWK1,SHR    DTSFILE EXTENT #1
// ASSGN SYS011,DISK,VOL=SYSWK2,SHR    DTSFILE EXTENT #2
// PAUSE ISSUE /DISC DTSFILE
// EXEC DTSUTIL
BACKUP
/*
// MTC RUN,SYS005
/&
* $$ EOJ
```

```
* $$ JOB JNM=EX421RIF,CLASS=0,DISP=D
// JOB EX421RIF  RESTORE SELECTED ICCF LIBRARIES
// TLBL DTSRSTR,'ICCF.BACKUP'
// ASSGN SYS004,1B0                INPUT TAPE (FROM BACKUP)
// MTC REW,SYS004
// ASSGN SYS010,DISK,VOL=SYSWK1,SHR    DTSFILE EXTENT #1
// ASSGN SYS011,DISK,VOL=SYSWK2,SHR    DTSFILE EXTENT #2
// PAUSE ISSUE /DISC DTSFILE
// EXEC DTSUTIL
RESTORE TO(21) FROM(21)
RESTORE TO(22) FROM(22)
RESTORE TO(23) FROM(23)
RESTORE TO(36) FROM(24)
RESTORE TO(37) FROM(25)
/*
// MTC RUN,SYS004
/&
* $$ EOJ
```

LABEL PROCEDURE SKELETON

DESCRIPTION :

THIS SAMPLE CONTAINS JOB CONTROL STATEMENTS TO CONSTRUCT
THE STANDARD LABELS FOR YOUR FILES AND DATA SETS.
IT ALSO CONTAINS THE POWER JECL AND THE JCL STATEMENTS TO
CATALOGUE THIS PROCEDURE INTO THE SYSTEM LIBRARY.

```
* $$ JOB JNM=CATALOG,DISP=D,CLASS=0
```

```
// JOB CATALOG
```

```
// EXEC LIBR
```

```
ACC S=IJSYSRS.SYSLIB
```

```
CATALOG STDLABUS.PROC DATA=YES REPLACE=YES
```

THE PROCEDURE NAME MUST BE STDLABUS AS THE
SYSTEM STANDARD LABEL PROCEDURE, STDLABEL, CALLS
STDLABUS TO LOAD YOUR STANDARD LABELS.

/. STANDARD LABEL SKELETON FOR YOUR STANDARD LABELS

SEQUENTIAL DISK FILE SAMPLES

SINGLE EXTENT SD FILE

```
*****  
* // DLBL SAMPSD1,'SAMPLE.SEQUENTL.DISK.FILE.ONE',1999/365,SD *  
* // EXTENT SYS004,SAMP01,1,0,2400,1200 *  
*****
```

TWO SINGLE EXTENT SD FILES USING SPLIT CYLINDER

```
*****  
* // DLBL SAMPSD2,'SAMPLE.SEQUENTL.DISK.FILE.TWO',1999/365,SD *  
* // EXTENT SYS004,SAMP01,8,0,2400,1200,5 *  
* // DLBL SAMPSD3,'SAMPLE.SEQUENTL.DISK.FILE.THREE',1999/365,SD *  
* // EXTENT SYS004,SAMP01,8,0,2406,1200,11 *  
*****
```

MULTI-EXTENT SD FILE

```
*****  
* // DLBL SAMPSD4,'SAMPLE.SEQUENTL.DISK.FILE.FOUR',1999/365,SD *  
* // EXTENT SYS004,SAMP01,1,0,2400,600 *  
* // EXTENT SYS004,SAMP01,1,1,2400,600 *  
*****
```

DIRECT ACCESS FILE SAMPLE

```
* * * * *
* // DLBL SAMPDA, 'SAMPLE.DIRECT.ACCESS.FILE.A', 1999/365, DA
* // EXTENT SYS002, SAMP01, 1, 0, 100, 1200
* // EXTENT SYS002, SAMP01, 1, 1, 1300, 1200
* * * * *
```

INDEX SEQUENTIAL FILE (CREATE) SAMPLE

```
* * * * *
* // DLBL SAMPISC, 'SAMPLE.INDEX.SEQUENTL.FILE.ONE', 1999/365, ISC
* // EXTENT SYS005, SAMP01, 4, 0, 3700, 1200
* // EXTENT SYS005, SAMP01, 4, 1, 4900, 1200
* // EXTENT SYS005, SAMP01, 1, 2, 6100, 1200
* // EXTENT SYS005, SAMP01, 1, 3, 7300, 1200
* // EXTENT SYS005, SAMP01, 2, 4, 8500, 1200
* * * * *
```

INDEX SEQUENTIAL (EXTENSION) FILE SAMPLE

```
* * * * *
* // DLBL SAMPISE, 'SAMPLE.INDEX.SEQUENTL.FILE.TWO', 1999/365, ISE
* // EXTENT SYS006, SAMP01, 4, 0, 9700, 1200
* // EXTENT SYS006, SAMP01, 4, 1, 10900, 1200
* // EXTENT SYS006, SAMP01, 1, 2, 12100, 1200
* // EXTENT SYS006, SAMP01, 1, 3, 13300, 1200
* // EXTENT SYS006, SAMP01, 2, 4, 14500, 1200
* * * * *
```

// OPTION PARSTD

ADD YOUR BG PARTITION STANDARD LABELS HERE IF REQUIRED.
IF NO LABELS ARE SUPPLIED THE AREA IS CLEARED.

// OPTION PARSTD=F1

ADD YOUR F1 PARTITION STANDARD LABELS HERE IF REQUIRED.
IF NO LABELS ARE SUPPLIED THE AREA IS CLEARED.

// OPTION PARSTD=F2

ADD YOUR F2 PARTITION STANDARD LABELS HERE IF REQUIRED.
IF NO LABELS ARE SUPPLIED THE AREA IS CLEARED.

// OPTION PARSTD=F3

ADD YOUR F3 PARTITION STANDARD LABELS HERE IF REQUIRED.
IF NO LABELS ARE SUPPLIED THE AREA IS CLEARED.

// OPTION PARSTD=F4

ADD YOUR F4 PARTITION STANDARD LABELS HERE IF REQUIRED.
IF NO LABELS ARE SUPPLIED THE AREA IS CLEARED.

// OPTION PARSTD=F5

ADD YOUR F5 PARTITION STANDARD LABELS HERE IF REQUIRED.
IF NO LABELS ARE SUPPLIED THE AREA IS CLEARED.

// OPTION PARSTD=F6

ADD YOUR F6 PARTITION STANDARD LABELS HERE IF REQUIRED.
IF NO LABELS ARE SUPPLIED THE AREA IS CLEARED.

// OPTION PARSTD=F7

ADD YOUR F7 PARTITION STANDARD LABELS HERE IF REQUIRED.
IF NO LABELS ARE SUPPLIED THE AREA IS CLEARED.

// OPTION PARSTD=F8

ADD YOUR F8 PARTITION STANDARD LABELS HERE IF REQUIRED.
IF NO LABELS ARE SUPPLIED THE AREA IS CLEARED.

// OPTION PARSTD=F9

ADD YOUR F9 PARTITION STANDARD LABELS HERE IF REQUIRED.
IF NO LABELS ARE SUPPLIED THE AREA IS CLEARED.

// OPTION PARSTD=FA

ADD YOUR FA PARTITION STANDARD LABELS HERE IF REQUIRED.
IF NO LABELS ARE SUPPLIED THE AREA IS CLEARED.

// OPTION PARSTD=FB

ADD YOUR FB PARTITION STANDARD LABELS HERE IF REQUIRED.
IF NO LABELS ARE SUPPLIED THE AREA IS CLEARED.

/+

/*

/&

* \$\$ EOJ

ASI MASTER PROCEDURE

```
* $$ JOB JNM=ASIPROC,CLASS=A,DISP=D, NTFY=YES, PRI=9
* $$ LST CLASS=R, DEST=(,MERZ), DISP=D
* $$ PUN CLASS=R, DEST=(,MERZ), DISP=D
// JOB ASIPROC CATALOG $ASI MASTER PROCEDURE
// EXEC LIBR,PARM='MSHP'
ACCESS SUBLIB=IJSYSRS.SYSLIB
CATALOG $ASIPROC.PROC      EOD=/+      REPLACE=YES
TYPE=NORMAL
CPU=FF0200064361,IPL=$IPL370,JCL=$$JCL370
CPU=020200064361,IPL=$IPLE,JCL=$$JCLE
CPU=FF0200224361,IPL=$IPLVAM,JCL=$$JCLVAM
CPU=FF0224084331,IPL=$IPLVMS,JCL=$$JCLVMS
CPU=FF0200444331,IPL=$IPLVUK,JCL=$$JCLVUK
/+
CONN S=IJSYSRS.SYSLIB:PRD2.SAVE
COPY $ASIPROC.PROC REPLACE=YES LIST=YES
/*
//EXEC LISTLOG
/&
* $$ EOJ
```

Note: Please refer to "Two Interactive Interfaces - Procedure Modifications" on page 384.

CONDITIONAL JOB CONTROL

Return Code Assembler Subroutines

The HLL languages do **not** provide the ability to pass return codes back to Job Control after program execution. However, this can be achieved by coding Assembler subroutines and therefore enabling the user to fully utilize the new conditional Job Control feature of VSE/SP 2.1.

An example of such a subroutine for PL/I and COBOL is shown below.

```
* $$ JOB JNM=RETCDE,CLASS=0,DISP=D
* $$ LST CLASS=R,DISP=D,DEST=(*,IANH)
* $$ PUN CLASS=R,DISP=D,DEST=(*,IANH)
// JOB RETCDE
// OPTION DECK
// EXEC ASSEMBLY,SIZE=128K
RETCDE   CSECT
        SAVE   (14,12)
        LR     5,15
        USING  RETCDE,5
        L      4,0(,1)          LOAD PARAMETER LIST ADDRESS
        PACK   RTCDP,0(4,4)      CONVERT FROM ZONED TO PACKED DEC.
        CVB    8,RTCDP          CONVERT PACKED DEC TO BINARY
        SR     15,15            CLEAR REG 15.
        LR     15,8             LOAD RETCODE INTO REG 15.
        EOJ    RC=(15)
        DS     0D
RTCDP    DS     PL8
        END

/*
/&
* $$ EOJ
```

In the above example, the user defined return code is passed as a parameter on the PL/I or COBOL statement. This value is passed to the subroutine as a **zoned decimal** field which is then converted to a **packed decimal** value. It is then converted to binary, loaded into general register 15, and then passed to Job Control via the EOJ macro.

Note: The EOJ macro halts program execution from the subroutine. Control is **not** passed back to the calling program. Therefore, before calling the subroutine, please ensure that any necessary end-of-job **housekeeping** is done from the main program.

Sample PL/I Program

The following is a sample PL/I program showing the statements necessary to call the Assembler subroutine. The return code is passed as a parameter via the **CALL** statement. In this example the **return code** passed will be **12**.

```
* $$ JOB JNM=PLITEST,CLASS=0,DISP=D
* $$ LST CLASS=R,DISP=D,DEST=(*,IANH)
// JOB PLITEST
// OPTION CATAL,LIST,SYM,NODECK,ERRS
  PHASE PLITEST,*
// EXEC PLIOPT,SIZE=128K
  MYPROG:PROC OPTIONS(MAIN);
  DECLARE
    1 RECAREA,
      2 (A,B,C,X1,X2) FLOAT DEC(6) COMPLEX;
  DCL RETCDE ENTRY OPTIONS(ASSEMBLER);
  DCL CODE PIC'9999' INIT ('0');
  ON ENDFILE (SYSIN) GOTO ENDJOB;
  OPEN FILE (SYSPRINT),
    FILE (SYSIN);
  LOOP:  GET FILE (SYSIN) LIST (A,B,C);
         PUT SKIP FILE (SYSPRINT) EDIT (RECAREA)(C(E(16,9)));
         GOTO LOOP;
  ENDJOB: CLOSE FILE(SYSPRINT),
    FILE (SYSIN);
  CODE = 12;
  CALL RETCDE (CODE);
  END MYPROG;

/*
// LIBDEF PHASE,CATALOG=ISCLIB.IANH
// LIBDEF *,SEARCH=ISCLIB.IANH
// EXEC LNKEDT,PARM='MSHP'
/*
// LIBDEF PHASE,SEARCH=ISCLIB.IANH
// OPTION PARTDUMP
// EXEC PLITEST
6 14 9
5 16 10
3 -12 2
/*
/&
* $$ EOJ
```

Sample COBOL Program

The following is a sample COBOL program showing the statements necessary to call the Assembler subroutine. The return code is passed as a parameter via the call statement. In this example the **return code** passed will be 5.

```
* $$ JOB JNM=CBLTEST,CLASS=0,DISP=D
* $$ LST CLASS=R,DISP=D,DEST=(*,IANH)
// JOB CBLTEST
// OPTION CATAL,LISTX,SYM,NODECK,ERRS
  PHASE CBLTEST,*
// EXEC FCOBOL,SIZE=128K
CBL NOLIB
  ID DIVISION.
  PROGRAM-ID. JACTEST.
  ENVIRONMENT DIVISION.
  INPUT-OUTPUT SECTION.
  FILE-CONTROL.
    SELECT INFILE ASSIGN SYS005-UR-2540R-S.
    SELECT PRTFILE ASSIGN SYS006-UR-1403-S.
  DATA DIVISION.
  FILE SECTION.
  FD INFILE LABEL RECORDS OMITTED.
  01 INREC PIC X(80).
  FD PRTFILE LABEL RECORDS OMITTED.
  01 PRTREC PIC X(80).
  WORKING-STORAGE SECTION.
  01 FILLER.
    02 RCODE PIC 9999.
  PROCEDURE DIVISION.
    MOVE ZERO TO RCODE.
    OPEN INPUT INFILE OUTPUT PRTFILE.
  MNLOOP.
    READ INFILE AT END GO TO ENDJOB.
    WRITE PRTREC FROM INREC.
    GO TO MNLOOP.
  ENDJOB.
    CLOSE INFILE PRTFILE.
    ADD 5 TO RCODE.
    CALL "RETCDE" USING RCODE.
  STOP RUN.

/*
// IF $RC GE 8 THEN
// GOTO ENDJOB
// LIBDEF PHASE,CATALOG=ISCLIB.IANH
// LIBDEF OBJ,SEARCH=ISCLIB.IANH
// EXEC LNKEDT,PARM='MSHP'
/*
// LIBDEF PHASE,SEARCH=(ISCLIB.IANH)
```

```

// ASSGN SYS006,SYSLST
// ASSGN SYS005,SYSIPT
// EXEC CBLTEST
FIRST RECORD
SECOND RECORD
THIRD RECORD
/*
/. ENDJOB
/&
* $$ EOJ

```

LINK PROCEDURES

VIRTUAL I/O

- Virtual I/O (VIO) is a system access method which uses an extension of the Page Data Set as its workspace instead of a normal workfile.
- Virtual I/O is currently only used by the **LINK** and **GO** functions of the Linkage Editor.
- Definition of the VIO workspace (during IPL together with the Supervisor name):
 - $VIO = n(K|M)$
 - Defines the size of the VIO workspace.
 - Default is 64K.
 - $VPOOL = m(K|M)$
 - Defines an area in the Virtual Address Space which provides access to the VIO workspace.
 - Default is 64K.
 - VPOOL is regarded as part of the VIO workspace.

VIO IMPLICATIONS

- Running MODE=VM
 - In this environment, VM/SP does the paging, therefore **no** Page Data Set exists on VSE/SP 2.1 and VIO is not used.
 - $VPOOL = nK$, where $n = (\text{no. of concurrent batch partitions} * \text{largest phase size})$
 - As the VPOOL resides at the end of the SVA, this may significantly reduce the size of virtual storage available for partitions.
- Running MODE=370
 - The Virtual Storage available for partitions is reduced by the size of the VPOOL.
 - For VAE systems, the available Virtual Storage in **each** address space is reduced by the size of the VPOOL.

USAGE OF OPTION CATAL

- Positive Aspects:
 - Multiphase/Overlay support
 - No reduction in available Virtual Storage
 - Can define a 'LINK' sublibrary where members are deleted after execution.
- Negative Aspects:
 - Requires JCL changes.
 - Need to code and submit a LIBR step to clean up the 'LINK' sublibrary.

Note: The 'negative aspects' can be overcome by coding a 'LINKGO' CMS EXEC or ICCF CLIST.

EXAMPLE OF A LINKGO ICCF CLIST

```
@MACRO
/SET DELAY BYPASS
@IF &&PARMCT NE 0 @FOR 2
@EXIT /ECHO NO PARAMETERS PASSED....ENTER 'LINKGO ?' FOR DETAILS.
/INP NOPROMPT
/LOAD DTSPROCS
/PARM &&PARAM1 &&PARAM2
/END
/ED
NEXT 2
I ==OPTIONS 0010011
I ==IF ==PARAM1 NE '?' ==GOTO NOTQUE
I ==TYPE THIS MACRO/PROCEDURE WILL BUILD AND SUBMIT A JOBSTREAM
I ==TYPE THAT SIMULATES THE 'LINK AND GO' FUNCTION.
I ==TYPE IT REQUIRES TWO PARAMETERS:
I ==TYPE      PARAM1 = THE ICCF MEMBER NAME OF THE SOURCE CODE.
I ==TYPE      PARAM2 = THE NAME OF THE COMPILER TO BE USED.
I ==EXIT
I ==LABEL NOTQUE
I ==SET ==VARBL1 ==PARAM1          SET MEMBER NAME
I ==SET ==VARBL2 ==PARAM2          SET COMPILER NAME
I ==SET ==VARBL5 ''
I ==SET ==VARBL3 ''
I ==TYPE DOES YOUR PROGRAM REQUIRE DLBLs AND ASSGNS?
I ==TYPE IF SO, PLEASE ENTER THE NAME OF THE ICCF MEMBER
I ==TYPE THAT CONTAINS THESE LABELS.
I ==TYPE IF NO LABELS REQUIRED...ENTER 'NONE'
I ==TYPE TO EXIT FROM THIS PROC ENTER 'QUIT'
I ==READ ==VARBL3
I ==IF ==VARBL3 EQ 'QUIT' ==EXIT
I ==IF ==VARBL3 NE 'NONE' ==GOTO GOTLAB
I ==SET ==VARBL4 '*'
I ==SET ==VARBL3 ''
I ==GOTO TSTIPT
I ==LABEL GOTLAB
I ==SET ==VARBL4 'INCLUDE'          SET LABELS MBR NAME
I ==SET ==VARBL5 'ICCFSLI'          FORCE * $$ SLI STMT
I ==LABEL TSTIPT
I ==TYPE DOES YOUR PROGRAM REQUIRE SYSIPT DATA?.
I ==TYPE IF SO, PLEASE ENTER THE NAME OF THE ICCF MEMBER
I ==TYPE THAT CONTAINS THIS DATA.
I ==TYPE IF NO SYSIPT DATA REQUIRED ..ENTER 'NONE'
I ==TYPE TO EXIT FROM THIS PROC ENTER 'QUIT'
I ==READ ==VARBL4
I ==IF ==VARBL4 EQ 'QUIT' ==EXIT
I ==IF ==VARBL4 NE 'NONE' ==GOTO GOTIPT
I ==SET ==VARBL4 ''
```

```

I ==SET ==VARBLB '*'
I ==GOTO TSTCOMP
I ==LABEL GOTIPT
I ==SET ==VARBLB 'INCLUDE'
I ==SET ==VARBLC 'ICCFSLI'
I ==LABEL TSTCOMP
I ==IF ==VARBL2 EQ 'ASSEMBLY' ==GOTO ASSEM
I ==IF ==VARBL2 EQ 'PLIOPT' ==GOTO PLIOPT
I ==IF ==VARBL2 EQ 'FCOBOL' ==GOTO COBOL
I ==IF ==VARBL2 EQ 'RPGII' ==GOTO RPG
I ==IF ==VARBL2 EQ 'VFORTTRAN' ==GOTO FORTR
I ==TYPE ==VARBL2 IS AN INVALID COMPILER
I ==TYPE VALID COMPILERS ARE:
I ==TYPE ASSEMBLY, FCOBOL, RPGII, VFORTTRAN AND PLIOPT.
I ==TYPE PLEASE ENTER THE CORRECT COMPILER OR 'QUIT'.
I ==READ ==VARBL2
I ==IF ==VARBL2 NE 'QUIT' ==GOTO -TSTCOMP
I ==EXIT
I ==LABEL PLIOPT
I ==SET ==VARBL6 '=128K'
I ==SET ==VARBL7 ',ERRS'
I ==SET ==VARBL8 ',LISTX'
I ==SET ==VARBL9 ',SYM'
I ==GOTO BUILD
I ==LABEL COBOL
I ==SET ==VARBL6 '=128K'
I ==SET ==VARBL7 ',ERRS'
I ==SET ==VARBL8 ',LISTX'
I ==SET ==VARBL9 ',SYM'
I ==GOTO BUILD
I ==LABEL RPG
I ==SET ==VARBL6 '=64K'
I ==GOTO BUILD
I ==LABEL FORTR
I ==SET ==VARBL6 '=700K'
I ==SET ==VARBL7 ',ERRS'
I ==SET ==VARBL8 ',LISTX'
I ==GOTO BUILD
I ==LABEL ASSEM
I ==SET ==VARBL6 '=128K'
I ==SET ==VARBL7 ',RLD'
I ==SET ==VARBL8 ',LIST'
I ==SET ==VARBL9 ',XREF'
I ==LABEL BUILD
I /INP NOPROMPT
I &/LOAD DTSSUBMT
I /UPSI 00000010
I /OPTION JSData
I /PARM ==USERIDLKGO DIRECT
I * $$ JOB JNM=DIRECT,CLASS=0,DISP=D,NTFY=YES
I * $$ LST CLASS=A,DISP=D,DEST=*
I * $$ EOJ
I XXXXXX
I // JOB ==VARBL1 SIMULATE LINK AND GO

```

```

I // ON $ABEND OR $CANCEL GOTO STEP4
I &/. STEP1
I // LIBDEF PHASE,CATALOG=TEST.LINK,TEMP
I // OPTION CATAL,NODECK==VARBL7==VARBL8==VARBL9
I   PHASE ==VARBL1,*
I // EXEC ==VARBL2,SIZE==VARBL6
I &/INCLUDE ==VARBL1 ICCFSLI
I &/*
I // IF $RC GE 12 THEN
I // GOTO $EOJ
I &/. STEP2
I // EXEC LNKEDT
I &/*
I &/. STEP3
I &==VARBLA ==VARBL3 ==VARBL5
I // LIBDEF PHASE,SEARCH=TEST.LINK,TEMP
I // EXEC ==VARBL1
I &==VARBLB ==VARBL4 ==VARBLC
I &/*
I &/. STEP4
I // EXEC LIBR
I A S=TEST.LINK
I DEL ==VARBL1.PHASE
I &/*
I &/&
I /END
I /PEND
I /RUN
TOP
C /==/&&/ * G
END
/RUN

```


EXAMPLE OF A LINKGO CMS EXEC

```

&TRACE OFF
CLRSCRN
&VSESYS = VSESP22
* ***** *
* THIS EXEC WILL BUILD AND SUBMIT A JOBSTREAM *
* THAT WILL SIMULATE THE 'LINK AND GO' *
* FUNCTION. *
* THE EXEC WILL ASK THE USER TO PROVIDE 4 *
* PARAMETERS: *
* *
* ==> PARAM1 = FN FT FM OF THE SOURCE CODE. M *
* ==> PARAM2 = NAME OF THE COMPILER. M *
* ==> PARAM3 = FN FT FM OF THE LABEL FILE. O *
* ==> PARAM4 = FN FT FM OF THE SYSIPT FILE. O *
* ***** *
*
* CHECK IF USER REQUIRES HELP.
*
&IF .&1 = .? &GOTO -HELP
*
* DETERMINE USER ID
*
USERID
&READ ARGS
&USER = &2
*
* ASK USER FOR SOURCE FILE.
*
-ASK1
&TYPE PLEASE ENTER THE FILENAME, FILETYPE AND (OPTIONALLY)
&TYPE THE FILEMODE OF THE CMS FILE THAT CONTAINS YOUR
&TYPE SOURCE CODE.
&TYPE IF NO FILEMODE IS SPECIFIED, A DEFAULT OF 'A' IS USED.
&TYPE
&TYPE ENTER FN FT (FM) OR 'QUIT' TO LEAVE THIS EXEC.
&READ ARGS
&IF .&1 = .QUIT &EXIT 10
&IF &INDEX LT 3 &FMS = A
&IF &INDEX EQ 3 &FMS = &3
&FNS = &1
&FTS = &2
STATE &FNS &FTS &FMS
&IF &RETCODE EQ 0 &GOTO -GOODS
CLRSCRN

```

```

&TYPE *
&TYPE * ==> THE FILE YOU SPECIFIED DOES NOT EXIST!!
&TYPE *
&GOTO -ASK1
-GOODS
*
* NOW WE WILL ASK FOR THE NAME OF THE COMPILER TO BE USED.
*
CLRSCRN
-ASKCOMP
&BEGTYPE -ENDTY
PLEASE ENTER THE NAME OF THE DOS/VSE COMPILER TO BE USED.
  VALID RESPONSES ARE:
                        ASSEMBLY
                        FCOBOL
                        RPGII
                        PLIOPT
                        VFORTRAN
  OR 'QUIT' TO LEAVE THIS EXEC.
-ENDTY
&READ ARGS
&IF .&1 = .QUIT &EXIT 10
&IF .&1 = .ASSEMBLY &GOTO -ASSEM
&IF .&1 = .PLIOPT &GOTO -PLI
&IF .&1 = .FCOBOL &GOTO -COBOL
&IF .&1 = .RPGII &GOTO -RPG
&IF .&1 = .VFORTRAN &GOTO -FORT
CLRSCRN
&TYPE *
&TYPE * YOU ENTERED AN INVALID COMPILER NAME!.
&TYPE *
&GOTO -ASKCOMP
*
* NOW WE WILL ASK FOR THE NAME OF THE CMS FILE FOR LABELS.
*
-ASKLABL
CLRSCRN
-ASKLABA
&BEGTYPE -ENDLB
PLEASE ENTER THE NAME OF THE CMS FILE THAT CONTAINS THE
NECESSARY LABELS/ASSGNS FOR YOUR PROGRAM.

==> ENTER FN FM (FT) -OR- 'NONE' IF NO LABELS REQUIRED
      OR 'QUIT' TO LEAVE THIS EXEC.
-ENDLB
&READ ARGS
&IF .&1 = .QUIT &EXIT 10
&IF .&1 = .NONE &GOTO -NOLAB
&IF &INDEX LT 3 &FML = A
&IF &INDEX EQ 3 &FML = &3
&FNL = &1
&FTL = &2
STATE &FNL &FTL &FML
&IF &RETCODE EQ 0 &GOTO -GOTLAB

```

```

CLRSCRN
&TYPE *
&TYPE * ==>  THE FILE YOU SPECIFIED DOES NOT EXIST!!
&TYPE *
&GOTO -ASKLABA
-NOLAB
&FNL = NONE
-GOTLAB
*
* NOW WE WILL ASK FOR THE NAME OF THE CMS FILE FOR SYSIPT.
*
-ASKIPTL
CLRSCRN
-ASKIPTA
&BEGTYPE -ENDIPT
PLEASE ENTER THE NAME OF THE CMS FILE THAT CONTAINS THE
NECESSARY SYSIPT DATA (IF ANY) FOR YOUR PROGRAM.

==>  ENTER FN FM (FT) -OR- 'NONE' IF NO SYSIPT REQUIRED
      OR 'QUIT' TO LEAVE THIS EXEC.
-ENDIPT
&READ ARGS
&IF .&1 = .QUIT &EXIT 10
&IF .&1 = .NONE &GOTO -NOIPT
&IF &INDEX LT 3 &FMI = A
&IF &INDEX EQ 3 &FMI = &3
&FNI = &1
&FTI = &2
STATE &FNI &FTI &FMI
&IF &RETCODE EQ 0 &GOTO -GOTIPT
CLRSCRN
&TYPE *
&TYPE * ==>  THE FILE YOU SPECIFIED DOES NOT EXIST!!
&TYPE *
&GOTO -ASKIPTA
-NOIPT
&FNI = NONE
&GOTO -GOTIPT
-PLI
&COMPN = PLIOPT
&SIZE = 128K
&OPT1 = ,ERRS
&OPT2 = ,LISTX
&OPT3 = ,SYM
&GOTO -ASKLABL
-COBOL
&COMPN = FCOBOL
&SIZE = 128K
&OPT1 = ,ERRS
&OPT2 = ,LISTX
&OPT3 = ,SYM
&GOTO -ASKLABL
-RPG
&COMPN = RPGII

```

```

&SIZE = 64K
&OPT1 =
&GOTO -ASKLABL
-FORT
&COMPN = VFORTRAN
&SIZE = 700K
&OPT1 = ,ERRS
&OPT2 = ,LISTX
&GOTO -ASKLABL
-ASSEM
&COMPN = ASSEMBLY
&SIZE = 128K
&OPT1 = ,RLD
&OPT2 = ,LIST
&OPT3 = ,XREF
&GOTO -ASKLABL
-GOTIPT
STATE XXXXXX JCLXXX A
&IF &RETCODE = 0 ERASE XXXXXX JCLXXX A
&STACK I * $$ JOB JNM=LINKGO,CLASS=0,DISP=D
&NAME = &CONCAT OF &USER )
&STACK I * $$ LST CLASS=R,DISP=D,DEST=(*,&NAME
&STACK I // JOB LINK AND GO
&STACK I // ON $ABEND OR $CANCEL GOTO STEP4
&STACK I /. STEP1
&STACK I // LIBDEF PHASE,CATALOG=TEST.LINK,TEMP
&OPTIONS = &CONCAT OF &OPT1 &OPT2 &OPT3
&STACK I // OPTION CATAL,NODECK&OPTIONS
&PHASE = &CONCAT OF &FNS ,*
&STACK I PHASE &PHASE
&EXECS = &CONCAT OF &COMPN ,SIZE= &SIZE
&STACK I // EXEC &EXECS
&STACK GET &FNS &FTS &FMS
&STACK I /*
&STACK I // IF $RC GE 12 THEN
&STACK I // GOTO $EOJ
&STACK I /. STEP2
&STACK I // EXEC LNKEDT
&STACK I /*
&STACK I /. STEP3
&IF &FNL = NONE &GOTO -MISLAB
&STACK GET &FNL &FTL &FML
-MISLAB
&STACK I // LIBDEF PHASE,SEARCH=TEST.LINK,TEMP
&STACK I // EXEC &FNS
&IF &FNI = NONE &GOTO -MISIPT
&STACK GET &FNI &FTI &FMI
-MISIPT
&STACK I /*
&STACK I /. STEP4
&STACK I // EXEC LIBR
&STACK I A S=TEST.LINK
&DELS = &CONCAT OF &FNS .PHASE
&STACK I DEL &DELS

```

```

&STACK I /*
&STACK I /&
&STACK I * $$ EOJ
&STACK TOP
&STACK C / / / * *
&STACK RECFM F
&STACK FILE
X XXXXXX JCLXXX A
CP SP PUN TO &VSESYS
PUNCH XXXXXX JCLXXX A (NOH
CP SP PUN FOR *
ERASE XXXXXX JCLXXX A
&EXIT
-HELP
&BEGTYPE -EHELP
* ***** *
* THIS EXEC WILL BUILD AND SUBMIT A JOBSTREAM *
* THAT WILL SIMULATE THE 'LINK AND GO' *
* FUNCTION. *
* THE EXEC WILL ASK THE USER TO PROVIDE 4 *
* PARAMETERS: *
* *
* ==> PARAM1 = FN FT FM OF THE SOURCE CODE. M *
* ==> PARAM2 = NAME OF THE COMPILER. M *
* ==> PARAM3 = FN FT FM OF THE LABEL FILE. O *
* ==> PARAM4 = FN FT FM OF THE SYSIPT FILE. O *
* ***** *
-EHELP
&EXIT 20

```

OUTPUT PRODUCED BY LINKGO CLIST

```
* $$ JOB JNM=HARRLKGO,CLASS=0,DISP=D
* $$ LST DISP=D,CLASS=A,DEST=*
// JOB MYSOURCE SIMULATE LINK AND GO
// ON $ABEND OR $CANCEL GOTO STEP4
/. STEP1
// LIBDEF PHASE,CATALOG=TEST.LINK,TEMP
// OPTION CATAL,NODECK,RLD,LIST,XREF
// PHASE MYSOURCE,*
// EXEC ASSEMBLY,SIZE=128K
* $$ SLI ICCF=(MYSOURCE),LIB=(0038,0002)
/*
// IF $RC GE 12 THEN
// GOTO $EOJ
/. STEP2
// EXEC LNKEDT
/*
/. STEP3
* $$ SLI ICCF=(MYLABEL),LIB=(0038,0002)
// LIBDEF PHASE,SEARCH=TEST.LINK,TEMP
// EXEC MYSOURCE
* $$ SLI ICCF=(MYIPT),LIB=(0038,0002)
/*
/. STEP4
// EXEC LIBR
A S=TEST.LINK
DEL MYSOURCE.PHASE
/*
/&
* $$ EOJ
```

CHAPTER 5. VSE LIBRARIES TRANSFER

INTRODUCTION

The entire chapter deals with the migration of Librarian jobs including job control statements which are related to the old Librarian.

An attempt was made to map all important functions of the old (pre-VSE/SP 2.1) Librarian to the new (LIBR) functions. For samples of a **one-to-one** mapping of the old and new functions, see "Librarian Jobstreams" on page 106. The sample jobs listed there do not make optimal use of the LIBR capabilities; they are intended to just show the connection.

This chapter:

- Summarizes the old Librarian functions.
- Compares the old and the new functions.
- Describes the DASD layouts and standard labels of the test environment used for all jobs documented in this chapter.
- Shows how the old libraries were mapped to the new libraries (sublibraries). It also documents the usage of the **Library Migration Table**.
- Maps the old and new Librarian functions via job samples. There was no attempt to optimize the new Librarian jobs.
- Describes the method of physically transferring libraries from the current system to VSE/SP 2.1.
- Discusses how to create and extend libraries in VSAM managed space.
- Describes in detail which job control and Librarian functions can be emulated or have to be (should be) adapted to the new environment.

GENERAL LIBRARIAN FUNCTIONS

ID	FUNCTION	SUBFUNCTION	PROGRAM
01	LIBRARY CREATION (DEFINITION)	CREATION OF NEW SYSRES CREATION OF PRIVATE LIBRARIES	CORGZ CORGZ
02	LIBRARY REORGANIZAT.	MERGE WITH COPY COMPARE LIBRARIES BACKUP LIBRARIES RESTORE LIBRARIES	CORGZ COPYSERV BACKUP RESTORE
03	LIBRARY MAINTENANCE	CATALOG PHASES MAINTAIN OTHER MEMBERS	LNKEDT MAINT
04	LIBRARY SERVICES	UPDATE DSPLY PUNCH DSPCH	xSERV x=D C R S P
05	E-MACRO MAINTENANCE AND SERVICES	UPDATE DSPLY PUNCH DSPCH	ESERV
06	NON-LIBRARIAN FUNCTIONS	— — — — —	LIBLIST LIBDROP LIBDEF PDZAP OBJMAINT

Note: The ID number in the following charts and job samples is used as a cross reference.

MAPPING OF OLD VS. NEW LIBRARIAN FUNCTIONS

Old Librarian Functions

OLD					
ID	REF. NB.	PROGRAM	CONTROL STATEMENT	UPSI	LIBDEF AND/OR ASSGN
01	001	CORGZ	ALLOC	—	ASSGN
	002	"	NEWVOL	—	NEW, ASSGN
02	003	CORGZ	MERGE / COPYx	—	FROM, TO, ASSGN
	004	COPYSERV	—	XXXXXXXXX	ASSGN
	005	BACKUP	BACKUP	—	ASSGN, SEARCH
	006	RESTORE	RESTORE	XXX	ASSGN, SEARCH
03	007	MAINT	CONDL / CONDS	—	TO, ASSGN
	008	LNKEDT	LINK	—	TO, ASSGN
	009	"	CATAL	—	TO, ASSGN
	010	MAINT	CATALx	—	TO, ASSGN
	011	"	RENAMx	—	TO, ASSGN
	012	"	UPDATE	—	TO, ASSGN
	013	"	DELETx	—	TO, ASSGN
04	014	DSERV	DSPLY, DSPLYS	—	FROM, ASSGN
	015	xSERV	DSPLY	—	FROM, ASSGN
	016	"	PUNCH	—	FROM, ASSGN
	017	"	DSPCH	—	FROM, ASSGN
05	018	ESERV		—	FROM, ASSGN

Note: Use ID and REF.NB. as cross reference.

New Librarian Functions

NEW					
ID	REF. NB.	PROGRAM	ACCESS / CONNECT (ONLY FOR MN.MT)	COMMAND	LIBDEF
01	001	LIBR	—	DEFINE	—
	002	"	—	DEFINE	—
02	003	LIBR	CONNECT	COPY	—
	004	"	CONNECT	COMPARE	—
	005	"	—	BACKUP	—
	006	"	—	RESTORE	—
03	007	—	—	—	—
	008	LNKEDT	—	LINK	—
	009	"	—	CATAL	CATALOG
	010	LIBR	ACCESS	CATALOG	—
	011	"	ACCESS	RENAME	—
	012	"	ACCESS	UPDATE	—
	013	"	ACCESS	DELETE	—
04	014	LIBR	ACCESS	LISTDIR	—
	015	"	ACCESS	LIST	—
	016	"	ACCESS	PUNCH	—
	017	"	ACCESS	L / PU	—
05	018	ESERV	—		SEARCH
	019	LIBR	—	CHANGE	—
	020	"	—	INPUT	—
	021	"	CONNECT	MOVE	—
	022	"	—	RELEASE	—
	023	"	—	TEST	—

DESCRIPTION OF DASD LAYOUTS

DASD Layout SIPO31

FILENAME	FILE-ID	SUB-LIBRARY	START BLOCK	NUMBER BLOCKS	END BLOCK
IJSYSRS	NEW.SYSRES.S31	—	2	35838	35839
S31UCL1	USER.CL.S31UCL1	—	35840	8960	44799
S31URL1	USER.RL.S31URL1	—	44800	3200	47999
S31USL1	USER.SL.S31USL1	—	48000	2240	50239
S31UPL1	USER.PL.S31UPL1	—	50240	1920	52159
S31UCL2	USER.CL.S31UCL2	—	52160	11200	63359
S31URL2	USER.RL.S31URL2	—	63360	4032	67391
S31USL2	USER.SL.S31USL2	—	67392	2816	70207
S31UPL2	USER.PL.S31UPL2	—	70208	2432	72639

DASD Layout SYSWK1 VSE/SP 2.1

FILENAME	FILE-ID	SUB-LIBRARY	START BLOCK	NUMBER BLOCKS	END BLOCK
SP2LIB	USER.LIBRARY.SP2LIB	USRSLB1 USRSLB2	509938	48000	95999

STDLABELS for SIPO31

```
*  
* LABELS FOR PRIVATE LIBRARIES SIPO 3.1  
*  
// DLBL S31UCL1,'USER.CL.S31UCL1',99/365,SD  
// EXTENT ,SIPO31,1,0,35840,8960  
// DLBL S31URL1,'USER.RL.S31URL1',99/365,SD  
// EXTENT ,SIPO31,1,0,44800,3200  
// DLBL S31USL1,'USER.SL.S31USL1',99/365,SD  
// EXTENT ,SIPO31,1,0,48000,2240  
// DLBL S31UPL1,'USER.PL.S31UPL1',99/365,SD  
// EXTENT ,SIPO31,1,0,50240,1920  
// DLBL S31UCL2,'USER.CL.S31UCL2',99/365,SD  
// EXTENT ,SIPO31,1,0,52160,11200  
// DLBL S31URL2,'USER.RL.S31URL2',99/365,SD  
// EXTENT ,SIPO31,1,0,63360,4032  
// DLBL S31USL2,'USER.SL.S31USL2',99/365,SD  
// EXTENT ,SIPO31,1,0,67392,2816  
// DLBL S31UPL2,'USER.PL.S31UPL2',99/365,SD  
// EXTENT ,SIPO31,1,0,70208,2432  
/*
```

STDLABELS for SYSWK1

```
*  
* LABELS FOR PRIVATE LIBRARIES VSE/SP 2.1  
*  
// DLBL SP2LIB,'USER.LIBRARY.SP2LIB',99/365,SD  
// EXTENT ,SYSWK1,1,0,509938,48000  
/*
```

MAPPING OF LIBRARIES

Library Migration Graphic

The graph shows how to transfer all old libraries with filename S31UxL1 to sublibrary SP2LIB.USRSLB1 and all old libraries with filename S31UxL2 to sublibrary SP2LIB.USRSLB2.

OLD		NEW			
FILE-NAME	FILE-ID	//	FILE-NAME	SUB-LIBRARY	FILE-ID
S31UCL1	USER.CL.S31UCL1	//	SP2LIB	USRSLB1	USER.LIBRARY.SP2-LIB
S31URL1	USER.RL.S31URL1	//			
S31USL1	USER.SL.S31USL1	//			
S31UPL1	USER.PL.S31UPL1	//			
S31UCL2	USER.CL.S31UCL2	//			
S31URL2	USER.RL.S31URL2	//		USRSLB2	
S31USL2	USER.SL.S31USL2	//			
S31UPL2	USER.PL.S31UPL2	//			
		//			

Create Library Migration Table

```
* $$ JOB JNM=MIGTABCR,CLASS=0,DISP=D
// JOB MIGTABCR  CREATE LIBRARY MIGRATION TABLE
* VSE/SP 2.1 MAPS SIPO 3.1 TO VSE/SP 2.1 LIBRARIES
// LIBDEF PHASE,CATALOG=SP2LIB.USRSLB1
// OPTION CATAL
// EXEC ASSEMBLY
    INLMIGR OLD=S31UCL1,LIB=SP2LIB,SUBLIB=USRSLB1
    INLMIGR OLD=S31URL1,LIB=SP2LIB,SUBLIB=USRSLB1
    INLMIGR OLD=S31USL1,LIB=SP2LIB,SUBLIB=USRSLB1
    INLMIGR OLD=S31UPL1,LIB=SP2LIB,SUBLIB=USRSLB1
    INLMIGR OLD=S31UCL2,LIB=SP2LIB,SUBLIB=USRSLB2
    INLMIGR OLD=S31URL2,LIB=SP2LIB,SUBLIB=USRSLB2
    INLMIGR OLD=S31USL2,LIB=SP2LIB,SUBLIB=USRSLB2
    INLMIGR OLD=S31UPL2,LIB=SP2LIB,SUBLIB=USRSLB2
END
/*
// EXEC LNKEDT
// LIBDEF PHASE,SEARCH=SP2LIB.USRSLB1
SET SDL
INLPLMT,SVA
/&
* $$ EOJ
```

Load Library Migration Table

```

* $$ JOB JNM=ASIZ2,CLASS=0,DISP=D
// JOB ASIZ2 CATALOG ASI PROCEDURE FOR 4331-2
// EXEC LIBR
ACCESS S=IJSYSRS.SYSLIB
DELETE $IPLSAV.PROC
RENAME $IPLZ2.PROC:$IPLSAV.=
CATALOG $IPLZ2.PROC,REPLACE=YES
01F,$$A$SUPB,P,LOG,VSIZE=32768K,VIO=1M,VPOOL=128K
ADD 12:17,3277 CICS/ICCF TERMINALS
ADD 1D,3540 USER DISKETTE
.
.
SVA SDL=256,PSIZE=768K,GETVIS=208K
/+
DELETE $OJCLSAV.PROC
RENAME $OJCLZ2.PROC:$OJCLSAV.=
CATALOG $OJCLZ2.PROC,REPLACE=YES
ALLOC S,F1=512K,F2=2M,F3=512K
ALLOC 1,FA=2M,FB=2M,BG=2M
.
.
/*
LIBDEF PHASE,SEARCH=(IJSYSRS.SYSLIB,PRODPPS.PR$AM2,PRODPPS.PR$AM4, C
PRODPPS.PR$SM2,SP2LIB.USRSLB1)

SET SDL
INLPLMT,SVA LIBRARY MIGRATION TABLE
$$BATTNA,MOVE SYSTEM, ATTEN ROUTINE
$$BATTNB,MOVE SYSTEM, ATTEN ROUTINE
$$BATTNH,MOVE SYSTEM, ATTEN ROUTINE
.
.
.
.
/*
/&
* $$ EOJ

```

LIBRARIAN JOBSTREAMS

Library Creation (Definition)

The job samples below show the most significant Librarian functions.

Each of the following pages is split into three distinct areas:

- Area 1** contains the job as coded for the old Librarian, that is, VSE SIPO/E 1.3.1. All parameters or statements that you must change are underscored.
- Area 2** contains the equivalent job for VSE/SP 2.1. Again, all important parameters are underscored.
- Area 3** shows how the individual parameters or statements or a combination thereof map from the old to the new environment.

Note that only the parameters or statements related to the Librarian or job control are given special attention. You must of course adapt others, such as volume-id or file-id to your system needs.

```
* $$ JOB JNM=S31L101,CLASS=0,DISP=D
// JOB S31L101 CREATE NEW SYSTEM RESIDENCE
* SIPO 3.1 LIBRARIAN ID=1 JOB=01
// DLBL IJSYSRS, 'NEW.SYSRES.S31', 99/365, SD
// EXTENT SYS002, SIPO31, 1, 0, 2, 35838
// ASSGN SYS002, DISK, VOL=SIPO31, SHR
// EXEC CORGZ
ALLOC CL=33662(128), RL=0(0), SL=0(0), PL=1920(64)
/*
/&
* $$ EOJ
```



```

* $$ JOB JNM=SP2L101,CLASS=0,DISP=D
// JOB SP2L101  CREATE NEW SYSTEM RESIDENCE
* VSE/SP 2.1 LIBRARIAN  ID=1  JOB=01
// DLBL IJSYSR1,'SYS.NEW.RES',99/365,SD
// EXTENT ,SYSWK1,1,0,2,55116
// EXEC LIBR
DEFine Lib=IJSYSR1
DEFine Sublib=IJSYSR1.SYSLIB
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

<u>IJSYSRS / SYS002</u>	=====>	<u>IJSYSR1</u>
<u>CORGZ / ALLOC</u>	=====]>	<u>DEF L=IJSYSR1</u>
	]>	<u>DEF S=IJSYSR1.SYSLIB</u>

```

* $$ JOB JNM=S31L102,CLASS=0,DISP=D
// JOB S31L102 CREATE PRIVATE LIBRARIES
* SIPO 3.1 LIBRARIAN ID=1 JOB=02
// LIBDEF CL,NEW=S31UCL2
// LIBDEF RL,NEW=S31URL2
// LIBDEF SL,NEW=S31USL2
// LIBDEF PL,NEW=S31UPL2
// EXEC CORGZ
NEWVOL CL=11200(128),RL=4032(64),SL=2816(64),PL=2432(64)
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L102,CLASS=0,DISP=D
// JOB SP2L102 CREATE PRIVATE LIBRARIES
* VSE/SP 2.1 LIBRARIAN ID=1 JOB=02
// EXEC LIBR
DEFine Lib=SP2LIB
DEFine Sublib=SP2LIB.USRSLB2
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

CL,NEW=S31UCL2	[=====]>	DEF L=SP2LIB
CL,NEW=S31URL2	[]>	DEF S=SP2LIB.USRSLB2
CL,NEW=S31USL2	[]	
CL,NEW=S31UPL2	[]	
CORGZ / NEWVOL	[]	

Library Reorganization

```
* $$ JOB JNM=S31L201,CLASS=0,DISP=D
// JOB S31L201 MERGE FROM SYSRES TO NEWRES WITH 'COPY ALL'
* SIPO 3.1 LIBRARIAN ID=2 JOB=01
// DLBL IJSYSRS,'NEW.SYSRES.S31',99/365,SD
// EXTENT SYS002,SIPO31,1,0,2,35838
// ASSGN SYS002,DISK,VOL=SIPO31,SHR
// EXEC CORGZ
MERGE RES,NRS
COPY ALL
/*
/&
* $$ EOJ
```

```
* $$ JOB JNM=SP2L201,CLASS=0,DISP=D
// JOB SP2L201 MERGE IJSYSRS TO NEWRES ALL
* VSE/SP 2.1. LIBRARIAN ID=2 JOB=01
// DLBL IJSYSR1,'SYS.NEW.RES',99/365,SD
// EXTENT ,SYSWK1,1,0,2,55116
// EXEC LIBR
COPY Lib=IJSYSRS:IJSYSR1
/*
/&
* $$ EOJ
```

MAPPING OF PARAMETERS

IJSYSRS / SYS002	=====>	IJSYSR1
CORGZ / MERGE RES,NRS	[=====>	COP LIB=IJSYSRS:IJSYSR1
COPY ALL	[	

```

* $$ JOB JNM=S31L202,CLASS=0,DISP=D
// JOB S31L202 MERGE FROM NEWRES TO SYSRES INDIVIDUAL PHASES
* SIPO 3.1 LIBRARIAN ID=2 JOB=02
// DLBL IJSYSRS,'NEW.SYSRES.S31',99/365,SD
// EXTENT SYS002,SIPO31,1,0,2,35838
// ASSGN SYS002,DISK,VOL=SIPO31,SHR
// EXEC CORGZ
  MERGE NRS,RES
  COPYC CLEANER
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L202,CLASS=0,DISP=D
// JOB SP2L202 MERGE FROM NEWRES TO SYSRES INDIVIDUAL PHASES
* VSE/SP 2.1 LIBRARIAN ID=2 JOB=02
// DLBL IJSYSR1,'SYS.NEW.RES',99/365,SD
// EXTENT ,SYSWK1,1,0,2,55116
// EXEC LIBR
CONnect Sublib=IJSYSR1.SYSLIB:IJSYSRS.SYSLIB
COPy CLEANER.PHASE Replace=Yes
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

IJSYSRS / SYS002	=====>	IJSYSR1
CORGZ / MERGE NRS,RES	=====>	CON S=IJSYSR1.SYSLIB: ...
COPYC CLEANER	=====>	COP CLEANER.PHASE R=Y

```

* $$ JOB JNM=S31L203,CLASS=0,DISP=D
// JOB S31L203  MERGE FROM PRIVATE TO SYRES INDIVIDUAL PROCEDURES
* SIPO 3.1 LIBRARIAN  ID=2  JOB=03
// LIBDEF PL,FROM=S31UPL1
// EXEC CORGZ
  MERGE PRV,RES
  COPYP CONDENSE,FASTFTCH
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L203,CLASS=0,DISP=D
// JOB SP2L203  MERGE FROM PRIVATE TO SYSRES PROCEDURES
* VSE/SP 2.1 LIBRARIAN  ID=2  JOB=03
// EXEC LIBR
CONnect Sublib=SP2LIB.USRSLB1:IJSYSRS.SYSLIB
COPy CONDENSE.PROC,FASTFTCH.PROC Replace=Yes
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

CORGZ / MERGE PRV,RES	[=====>	CON S=SP2LIB.USRSLB1: ...
PL,FROM=S31UPL1	[	
COPYP CONDENSE, ...	=====>	COP CONDENSE.PROC, ...

```

* $$ JOB JNM=S31L204,CLASS=0,DISP=D
// JOB S31L204  MERGE FROM SYSRES TO NEWRES WITH 'COPY ALL'
* SIPO 3.1 LIBRARIAN ID=2  JOB=04
// DLBL IJSYSRS,'NEW.SYSRES.S31',99/365,SD
// EXTENT SYS002,SIPO31,1,0,2,35838
// ASSGN SYS002,DISK,VOL=SIPO31,SHR
// EXEC CORGZ
    MERGE RES,NRS
    COPY ALL
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L204,CLASS=0,DISP=D
// JOB SP2L204  MERGE FROM SYSRES TO NEWRES ALL PROCEDURES
* VSE/SP 2.1 LIBRARIAN ID=2  JOB=04
// DLBL IJSYSR1,'SYS.NEW.RES',99/365,SD
// EXTENT ,SYSWK1,1,0,2,55116
// EXEC LIBR
CONnect Sublib=IJSYSRS.SYSLIB:IJSYSR1.SYSLIB
COPy *.PROC Replace=Yes
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

<u>IJSYSRS / SYS002</u>	=====>	<u>IJSYSR1</u>
<u>CORGZ / MERGE RES,NRS</u>	=====>	<u>CON S=IJSYSRS.SYSLIB: ...</u>
<u>COPY ALL</u>	=====>	<u>COP *.PROC R=Y</u>

```

* $$ JOB JNM=S31L205,CLASS=0,DISP=D
// JOB S31L205 MERGE FROM SYSRES TO PRIVATE WITH 'COPYC XXXX.ALL'
* SIPO 3.1 LIBRARIAN ID=2 JOB=05
// UPSI 1
// LIBDEF CL,TO=S31UCL1
// EXEC CORGZ
MERGE RES,PRV
COPYC ASSE.ALL
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L205,CLASS=0,DISP=D
// JOB SP2L205 MERGE FROM SYSRES TO PRIVATE WITH COPY ASSE*.PHASE
* VSE/SP 2.1 LIBRARIAN ID=2 JOB=05
// EXEC LIBR
CONnect Sublib=IJSYSRS.SYSLIB:SP2LIB.USRSLB1
COPY ASSE*.PHASE Replace=Yes LISt=Yes
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

CORGZ / MERGE RES,PRV	[=====>	CON S=IJSYSRS.SYSLIB: ...
CL,TO=S31UCL1	[	
COPYC ASSE.ALL	=====>	COP ASSE*.PHASE
UPSI 1	=====>	LIS=Y

```

* $$ JOB JNM=S31L206,CLASS=0,DISP=D
// JOB S31L206 MERGE FROM NEWRES TO PRIVATE WITH 'COPYP NEW'
* SIPO 3.1 LIBRARIAN ID=2 JOB=06
// UPSI 1
// DLBL IJSYSRS,'NEW.SYSRES.S31',99/365,SD
// EXTENT SYS002,SIPO31,1,0,2,35838
// ASSGN SYS002,DISK,VOL=SIPO31,SHR
// LIBDEF PL,TO=S31UPL1
// EXEC CORGZ
MERGE NRS,PRV
COPYP NEW
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L206,CLASS=0,DISP=D
// JOB SP2L206 MERGE FROM NEWRES TO PRIVATE ALL NEW PROCS
* VSE/SP 2.1 LIBRARIAN ID=2 JOB=06
// DLBL IJSYSR1,'SYS.NEW.RES',99/365,SD
// EXTENT ,SYSWK1,1,0,2,55116
// EXEC LIBR
CONnect Sublib=IJSYSR1.SYSLIB:SP2LIB.USRSLB1
COPy *.PROC Replace=No LISt=Yes
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

IJSYSRS / SYS002	=====>	IJSYSR1
CORGZ / MERGE NRS,PRV	=====>	CON S=IJSYSR1.SYSLIB: ...
COPYP NEW	=====>	COP *.PROC R=N
UPSI 1	=====>	LIS=Y


```

* $$ JOB JNM=S31L207,CLASS=0,DISP=D
// JOB S31L207 MERGE FROM PRIVATE TO NEWRES WITH 'COPYC ALL'
* SIPO 3.1 LIBRARIAN ID=2 JOB=07
// DLBL IJSYSRS,'NEW.SYSRES.S31'99/365,SD
// EXTENT SYS002,SIPO31,1,0,2,35838
// ASSGN SYS002,DISK,VOL=SIPO31,SHR
// LIBDEF CL,FROM=S31UCL1
// EXEC CORGZ
MERGE PRV,NRS
COPYC ALL
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L207,CLASS=0,DISP=D
// JOB SP2L207 MERGE FROM PRIVATE TO NEWRES WITH COPY *.PHASE
* VSE/SP 2.1 LIBRARIAN ID=2 JOB=07
// DLBL IJSYSR1,'SYS.NEW.RES',99/365,SD
// EXTENT ,SYSWK1,1,0,2,55116
// EXEC LIBR
CONnect Sublib=SP2LIB.USRSLB1:IJSYSR1.SYSLIB
COPy *.PHASE Replace=Yes
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

CORGZ / MERGE PRV,NRS	[=====]>	IJSYSR1
CL,FROM=S31UCL1	[]>	CON S=SP2LIB.USRSLB1: ...
IJSYSRS / SYS002	[]	
COPYC ALL	=====>	COP *.PHASE

```

* $$ JOB JNM=S31L208,CLASS=0,DISP=D
// JOB S31L208 MERGE FROM PRIVATE TO PRIVATE WITH 'COPYC NEW'
* SIPO 3.1 LIBRARIAN ID=2 JOB=08
// UPSI 1
// LIBDEF CL,FROM=S31UCL1,TO=S31UCL2
// EXEC CORGZ
MERGE PRV,PRV
COPYC NEW
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L208,CLASS=0,DISP=D
// JOB SP2L208 COPY FROM PRIVATE TO PRIVATE *.PHASE Replace=NO
* VSE/SP 2.1 LIBRARIAN ID=2 JOB=08
// EXEC LIBR
CONnect Sublib=SP2LIB.USRSLB1:=.USRSLB2
COPY *.PHASE Replace=No LISt=Yes
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

CORGZ / MERGE PRV,PRV	[=====>	CON S=SP2LIB.USRSLB1: ...
CL,FROM=...,TO=...	[	
COPYC NEW	=====>	COP *.PHASE R=N
UPSI 1	=====>	LIS=Y

```

* $$ JOB JNM=S31L209,CLASS=0,DISP=D
// JOB S31L209 MERGE FROM PRIVATE TO PRIVATE WITH 'COPYR MN,MN,...'
* SIPO 3.1 LIBRARIAN ID=2 JOB=09
// LIBDEF RL,FROM=S31URL1,TO=S31URL2
// EXEC CORGZ
MERGE PRV,PRV
COPYR IJDPR3,IJBTVTN,IJB116CF
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L209,CLASS=0,DISP=D
// JOB SP2L209 MERGE FROM PRIVATE TO PRIVATE INDIVIDUAL OBJECTS
* VSE/SP 2.1 LIBRARIAN ID=2 JOB=09
// EXEC LIBR
CONnect Sublib=SP2LIB.USRSLB1:=.USRSLB2
COPy IJDPR3.OBJ,IJBTVTN.OBJ,IJB116CF.OBJ Replace=Yes
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

CORGZ / MERGE PRV,PRV	[=====>	CON S=SP2LIB.USRSLB1: ...
RL,FROM=...,TO=...	[	
COPYR IJDPR3	=====>	COP IJDPR3.OBJ

```

* $$ JOB JNM=S31L210,CLASS=0,DISP=D
// JOB S31L210 MERGE FROM PRIVATE TO PRIVATE WITH 'COPYR XXX.ALL'
* SIPO 3.1 LIBRARIAN ID=2 JOB=10
// UPSI 1
// LIBDEF RL,FROM=S31URL1,TO=S31URL2
// EXEC CORGZ
MERGE PRV,PRV
COPYR IJF.ALL
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L210,CLASS=0,DISP=D
// JOB SP2L210 MERGE FROM PRIVATE TO PRIVATE WITH IJF*.OBJ
* VSE/SP 2.1 LIBRARIAN ID=2 JOB=10
// EXEC LIBR
CONnect Sublib=SP2LIB.USRSLB1:=.USRSLB2
COPy IJF*.OBJ Replace=Yes LISt=YES
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

CORGZ / MERGE PRV,PRV	[=====>	CON S=SP2LIB.USRSLB1: ...
RL,FROM=...,TO=...	[	
COPYR IJF.ALL	=====>	COP IJF*.OBJ
UPSI 1	=====>	LIS=Y

```

* $$ JOB JNM=S31L211,CLASS=0,DISP=D
// JOB S31L211 MERGE FROM PRIVATE TO PRIVATE WITH 'COPYR NEW'
* SIPO 3.1 LIBRARIAN ID=2 JOB=11
// UPSI 1
// LIBDEF RL,FROM=S31URL1,TO=S31URL2
// EXEC CORGZ
MERGE PRV,PRV
COPYR NEW
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L211,CLASS=0,DISP=D
// JOB SP2L211 MERGE FROM PRIVATE TO PRIVATE WITH NEW OBJECTS
* VSE/SP 2.1 LIBRARIAN ID=2 JOB=11
// EXEC LIBR
CONnect Sublib=SP2LIB.USRSLB1:=.USRSLB2
COPY *.OBJ Replace=No LISt=Yes
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

CORGZ / MERGE PRV,PRV	[=====>	CON S=SP2LIB.USRSLB1: ...
RL,FROM=...,TO=...	[	
COPYR NEW	=====>	COP *.OBJ R=N
UPSI 1	=====>	LIS=Y

```

* $$ JOB JNM=S31L212,CLASS=0,DISP=D
// JOB S31L212 MERGE FROM PRIVATE TO PRIVATE WITH 'COPYR ALL'
* SIPO 3.1 LIBRARIAN ID=2 JOB=12
// LIBDEF RL,FROM=S31URL1,TO=S31URL2
// EXEC CORGZ
MERGE PRV,PRV
COPYR ALL
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L212,CLASS=0,DISP=D
// JOB SP2L212 MERGE FROM PRIVATE TO PRIVATE ALL OBJECTS
* VSE/SP 2.1 LIBRARIAN ID=2 JOB=12
// EXEC LIBR
CONnect Sublib=SP2LIB.USRSLB1:=.USRSLB2
COPy *.OBJ Replace=Yes
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

CORGZ / MERGE PRV,PRV	[=====>	CON S=SP2LIB.USRSLB1: ...
RL,FROM=...,TO=...	[	
COPYR ALL	=====>	COP *.OBJ

```

* $$ JOB JNM=S31L213,CLASS=0,DISP=D
// JOB S31L213 MERGE FROM PRIVATE TO PRIVATE WITH 'COPYS SB,SB,...'
* SIPO 3.1 LIBRARIAN ID=2 JOB=13
// LIBDEF SL,FROM=S31USL1,TO=S31USL2
// EXEC CORGZ
MERGE PRV,PRV
COPYS A.RJEVERF,B.APP,E.PUT
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L213,CLASS=0,DISP=D
// JOB SP2L213 MERGE PRIVATE TO PRIVATE CERTAIN SOURCE BOOKS
* VSE/SP 2.1 LIBRARIAN ID=2 JOB=13
// EXEC LIBR
CONnect Sublib=SP2LIB.USRSLB1:=.USRSLB2
COPy RJEVERF.A,APP.B,PUT.E Replace=Yes
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

CORGZ / MERGE PRV,PRV	[=====>	CON S=SP2LIB.USRSLB1: ...
SL,FROM=...,TO=...	[	
COPYS A.RJEVERF	=====>	COP RJEVERF.A

```

* $$ JOB JNM=S31L214,CLASS=0,DISP=D
// JOB S31L214 MERGE FROM PRIVATE TO PRIVATE WITH 'COPYS X.ALL'
* SIPO 3.1 LIBRARIAN ID=2 JOB=14
// UPSI 1
// LIBDEF SL,FROM=S31USL1,TO=S31USL2
// EXEC CORGZ
MERGE PRV,PRV
COPYS B.ALL
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L214,CLASS=0,DISP=D
// JOB SP2L214 MERGE PRIVATE TO PRIVATE ALL B. SOURCE BOOKS
* VSE/SP 2.1 LIBRARIAN ID=2 JOB=14
// EXEC LIBR
CONnect Sublib=SP2LIB.USRSLB1:=.USRSLB2
COPy *.B Replace=Yes LIST=YES
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

CORGZ / MERGE PRV,PRV	[=====>	CON S=SP2LIB.USRSLB1: ...
SL,FROM=...,TO=...	[	
COPYS B.ALL	=====>	COP *.B
UPSI 1	=====>	LIS=Y


```

* $$ JOB JNM=S31L215,CLASS=0,DISP=D
// JOB S31L215 MERGE FROM PRIVATE TO PRIVATE WITH 'COPYS NEW'
* SIPO 3.1 LIBRARIAN ID=2 JOB=15
// UPSI 1
// LIBDEF SL,FROM=S31USL1,TO=S31USL2
// EXEC CORGZ
MERGE PRV,PRV
COPYS NEW
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L215,CLASS=0,DISP=D
// JOB SP2L215 MERGE FROM PRIVATE TO PRIVATE ALL NEW SOURCE BOOKS
* VSE/SP 2.1 LIBRARIAN ID=2 JOB=15
// EXEC LIBR
CONnect Sublib=SP2LIB.USRSLB1:=.USRSLB2
COPy *.A,*.B,*.E Replace=No LISt=Yes
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

CORGZ / MERGE PRV,PRV	[=====>	CON S=SP2LIB.USRSLB1: ...
SL,FROM=...,TO=...	[	
COPYS NEW	=====>	COP *.A,*.B,*.E R=N
UPSI 1	=====>	LIS=Y

```

* $$ JOB JNM=S31L216,CLASS=0,DISP=D
// JOB S31L216 MERGE FROM PRIVATE TO PRIVATE WITH 'COPYS ALL'
* SIPO 3.1 LIBRARIAN ID=2 JOB=16
// LIBDEF SL,FROM=S31USL1,TO=S31USL2
// EXEC CORGZ
MERGE PRV,PRV
COPYS ALL
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L216,CLASS=0,DISP=D
// JOB SP2L216 MERGE PRIVATE TO PRIVATE ALL SOURCE BOOKS
* VSE/SP 2.1 LIBRARIAN ID=2 JOB=16
// EXEC LIBR
CONnect Sublib=SP2LIB.USRSLB1:=.USRSLB2
COPy *.A,*.B,*.E Replace=Yes
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

CORGZ / MERGE PRV,PRV	[=====>	CON S=SP2LIB.USRSLB1: ...
SL,FROM=...,TO=...	[	
COPYS ALL	=====>	COP *.A,*.B,*.E

```

* $$ JOB JNM=S31L220,CLASS=0,DISP=D
// JOB S31L220 COMPARE SYSRES WITH NEWRES
* SIPO 3.1 LIBRARIAN ID=2 JOB=20
* COPYSERV WILL ONLY RUN ON CKD DEVICES
// UPSI OXXXOXXX 0=SYSRES ----0=NEWRES
// DLBL IJSYSNR,'NEW.SYSRES.S31.CKD',99/365,SD
// EXTENT SYS003,SIPO31,1,0,1,1799
// ASSGN SYS003,DISK,VOL=SIPO31,SHR
// EXEC COPYSERV
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L220,CLASS=0,DISP=D
// JOB SP2L220 COMPARE SYSRES WITH NEWRES
* VSE/SP 2.1 LIBRARIAN ID=2 JOB=20
// DLBL IJSYSR1,'SYS.NEW.RES',99/365,SD
// EXTENT ,SYSWK1,1,0,2,55116
// EXEC LIBR
COMpare Lib=IJSYSRS:IJSYSR1
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

COPYSERV	=====>	COM
UPSI 0	=====>	L=IJSYSRS
UPSI ----0	[=====>	... : IJSYSR1
IJSYSNR / SYS003	[	

```

* $$ JOB JNM=S31L221,CLASS=0,DISP=D
// JOB S31L221 COMPARE SYSRES WITH PRIVATE CIL
* SIPO 3.1 LIBRARIAN ID=2 JOB=21
* COPYSERV WILL ONLY RUN ON CKD DEVICES
// UPSI 0XXX1XXX 0=SYSRES ----1=PRIVATE CIL
// DLBL IJSYSNR,'USER.CL.S31UCL1.CKD',99/365,SD
// EXTENT SYS003,SIPO31,1,0,1800,720
// ASSGN SYS003,DISK,VOL=SIPO31,SHR
// EXEC COPYSERV
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L221,CLASS=0,DISP=D
// JOB SP2L221 COMPARE SYSRES WITH PRIVATE PHASES
* VSE/SP 2.1 LIBRARIAN ID=2 JOB=21
// EXEC LIBR
CONnect Sublib=IJSYSRS.SYSLIB:SP2LIB.USRSLB1
COMpare *.PHASE
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

COPYSERV	=====>	COM *.PHASE
UPSI 0	=====>	CON S=IJSYSRS.SYSLIB
UPSI ----1	[=====>	... : SP2LIB.USRSLB1
IJSYSNR / SYS003	[	

```

* $$ JOB JNM=S31L222,CLASS=0,DISP=D
// JOB S31L222 COMPARE SYSRES WITH PRIVATE RELO
* SIPO 3.1 LIBRARIAN ID=2 JOB=22
* COPYSERV WILL ONLY RUN ON CKD DEVICES
// UPSI OXXXX1XX 0=SYSRES -----1=PRIVATE RELO
// DLBL IJSYSNR,'USER.RL.S31URL1.CKD',99/365,SD
// EXTENT SYS003,SIPO31,1,0,2520,720
// ASSGN SYS003,DISK,VOL=SIPO31,SHR
// EXEC COPYSERV
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L222,CLASS=0,DISP=D
// JOB SP2L222 COMPARE SYSRES WITH PRIVATE OBJECTS
* VSE/SP 2.1 LIBRARIAN ID=2 JOB=22
// EXEC LIBR
CONnect Sublib=IJSYSRS.SYSLIB:SP2LIB.USRSLB1
COMpare *.OBJ
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

COPYSERV	=====>	COM *.OBJ
UPSI 0	=====>	CON S=IJSYSRS.SYSLIB
UPSI -----1	[=====>	... : SP2LIB.USRSLB1
IJSYSNR / SYS003	[	

```

* $$ JOB JNM=S31L223,CLASS=0,DISP=D
// JOB S31L223 COMPARE SYSRES WITH PRIVATE SOURCE
* SIPO 3.1 LIBRARIAN ID=2 JOB=23
* COPYSERV WILL ONLY RUN ON CKD DEVICES
// UPSI 0XXXXX1X 0=SYSRES -----1=PRIVATE SOURCE
// DLBL IJSYSNR,'USER.SL.S31USL1.CKD',99/365,SD
// EXTENT SYS003,SIPO31,1,0,3240,720
// ASSGN SYS003,DISK,VOL=SIPO31,SHR
// EXEC COPYSERV
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L223,CLASS=0,DISP=D
// JOB SP2L223 COMPARE SYSRES WITH PRIVATE SOURCE BOOKS
* VSE/SP 2.1 LIBRARIAN ID=2 JOB=23
// EXEC LIBR
CONnect Sublib=IJSYSRS.SYSLIB:SP2LIB.USRSLB1
COMpare *.A,*.B,*.E
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

COPYSERV	=====>	COM *.a,*.b,*.e
UPSI 0	=====>	CON S=IJSYSRS.SYSLIB
UPSI -----1	[=====>	... : SP2LIB.USRSLB1
IJSYSNR / SYS003	[	

```

* $$ JOB JNM=S31L224,CLASS=0,DISP=D
// JOB S31L224 COMPARE PRIVATE CIL WITH NEWRES
* SIPO 3.1 LIBRARIAN ID=2 JOB=24
* COPYSERV WILL ONLY RUN ON CKD DEVICES
// UPSI 1XXX0XXX 1=PRIVATE CIL ----0=NEWRES
// DLBL IJSYSCL,'USER.CL.S31UCL1.CKD',99/365,SD
// EXTENT SYSCLB,SIPO31,1,0,1800,720
ASSGN SYSCLB,DISK,VOL=SIPO31,SHR
// DLBL IJSYSNR,'NEW.SYSRES.S31.CKD',99/365,SD
// EXTENT SYS003,SIPO31,1,0,1,1799
// ASSGN SYS003,DISK,VOL=SIPO31,SHR
// EXEC COPYSERV
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L224,CLASS=0,DISP=D
// JOB SP2L224 COMPARE PRIVATE PHASES WITH NEWRES PHASES
* VSE/SP 2.1 LIBRARIAN ID=2 JOB=24
// DLBL IJSYSR1,'SYS.NEW.RES',99/365,SD
// EXTENT ,SYSWK1,1,0,2,55116
// EXEC LIBR
CONnect Sublib=SP2LIB.USRSLB1:IJSYSR1.SYSLIB
COMpare *.PHASE
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

COPYSERV	=====>	COM *.PHASE
UPSI 1	[=====>	CON S=SP2LIB.USRSLB1
IJSYSCL / SYSCLB	[	
UPSI ----0	[=====>	... :IJSYSR1.SYSLIB
IJSYSNR / SYS003	[	

```

* $$ JOB JNM=S31L225,CLASS=0,DISP=D
// JOB S31L225 COMPARE PRIVATE RELO WITH NEWRES
* SIPO 3.1 LIBRARIAN ID=2 JOB=25
* COPYSERV WILL ONLY RUN ON CKD DEVICES
// UPSI X1XX0XXX -1=PRIVATE RELO ----0=NEWRES
// DLBL IJSYSRL,'USER.RL.S31URL1.CKD',99/365,SD
// EXTENT SYSRLB,SIPO31,1,0,2520,720
// ASSGN SYSRLB,DISK,VOL=SIPO31,SHR
// DLBL IJSYSNR,'NEW.SYSRES.S31.CKD',99/365,SD
// EXTENT SYS003,SIPO31,1,0,1,1799
// ASSGN SYS003,DISK,VOL=SIPO31,SHR
// EXEC COPYSERV
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L225,CLASS=0,DISP=D
// JOB SP2L225 COMPARE PRIVATE OBJECTS WITH NEWRES
* VSE/SP 2.1 LIBRARIAN ID=2 JOB=25
// DLBL IJSYSR1,'SYS.NEW.RES',99/365,SD
// EXTENT ,SYSWK1,1,0,2,55116
// EXEC LIBR
CONnect Sublib=SP2LIB.USRSLB1:IJSYSR1.SYSLIB
COMpare *.OBJ
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

COPYSERV	=====>	COM *.OBJ
UPSI -1	[=====>	CON S=SP2LIB.USRSLB1
IJSYSRL / SYSRLB	[	
UPSI ----0	[=====>	... :IJSYSR1.SYSLIB
IJSYSNR / SYS003	[	


```

* $$ JOB JNM=S31L226,CLASS=0,DISP=D
// JOB S31L226 COMPARE PRIVATE SOURCE WITH NEWRES
* SIPO 3.1 LIBRARIAN ID=2 JOB=26
* COPYSERV WILL ONLY RUN ON CKD DEVICES
// UPSI XX1X0XXX --1=PRIVATE SOURCE ----0=NEWRES
// DLBL IJSYSSL,'USER.SL.S31USL1.CKD',99/365,SD
// EXTENT SYSSLB,SIPO31,1,0,3240,720
// ASSGN SYSSLB,DISK,VOL=SIPO31,SHR
// DLBL IJSYSNR,'NEW.SYSRES.S31.CKD',99/365,SD
// EXTENT SYS003,SIPO31,1,0,1,1799
// ASSGN SYS003,DISK,VOL=SIPO31,SHR
// EXEC COPYSERV
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L226,CLASS=0,DISP=D
// JOB SP2L226 COMPARE PRIVATE SOURCE BOOKS WITH NEWRES
* VSE/SP 2.1 LIBRARIAN ID=2 JOB=26
// DLBL IJSYSR1,'SYS.NEW.RES',99/365,SD
// EXTENT ,SYSWK1,1,0,2,55116
// EXEC LIBR
CONnect Sublib=SP2LIB.USRSLB1:IJSYSR1.SYSLIB
COMpare *.A,*.B,*.E
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

COPYSERV	=====>	COM *.A,*.B,*.E
UPSI --1	[=====>	CON S=SP2LIB.USRSLB1
IJSYSSL / SYSSLB	[	
UPSI ----0	[=====>	... :IJSYSR1.SYSLIB
IJSYSNR / SYS003	[	

```

* $$ JOB JNM=S31L227,CLASS=0,DISP=D
// JOB S31L227 COMPARE PRIVATE CIL WITH PRIVATE CIL
* SIPO 3.1 LIBRARIAN ID=2 JOB=27
* COPYSERV WILL ONLY RUN ON CKD DEVICES
// UPSI 1XXX1XXX 1=PRIVATE CIL ----1=PRIVATE CIL
// DLBL IJSYSCL,'USER.CL.S31UCL1.CKD',99/365,SD
// EXTENT SYSCLB,SIPO31,1,0,1800,720
ASSGN SYSCLB,DISK,VOL=SIPO31,SHR
// DLBL IJSYSNR,'USER.CL.S31UCL2.CKD',99/365,SD
// EXTENT SYS003,SIPO31,1,0,4680,720
// ASSGN SYS003,DISK,VOL=SIPO31,SHR
// EXEC COPYSERV
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L227,CLASS=0,DISP=D
// JOB SP2L227 COMPARE PRIVATE PHASES WITH PRIVATE PHASES
* VSE/SP 2.1 LIBRARIAN ID=2 JOB=27
// EXEC LIBR
CONnect Sublib=SP2LIB.USRSLB1:=.USRSLB2
COMpare *.PHASE
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

COPYSERV	=====>	COM *.PHASE
UPSI 1	[=====>	CON S=SP2LIB.USRSLB1
IJSYSCL / SYSCLB	[	
UPSI ----1	[=====>	... :=.USRSLB2
IJSYSNR / SYS003	[	

```

* $$ JOB JNM=S31L228,CLASS=0,DISP=D
// JOB S31L228 COMPARE PRIVATE RELO WITH PRIVATE RELO
* SIPO 3.1 LIBRARIAN ID=2 JOB=28
* COPYSERV WILL ONLY RUN ON CKD DEVICES
// UPSI X1XXX1XX -1=PRIVATE RELO -----1=PRIVATE RELO
// DLBL IJSYSRL,'USER.RL.S31URL1.CKD',99/365,SD
// EXTENT SYSRLB,SIPO31,1,0,2520,720
// ASSGN SYSRLB,DISK,VOL=SIPO31,SHR
// DLBL IJSYSNR,'USER.RL.S31URL2.CKD',99/365,SD
// EXTENT SYS003,SIPO31,1,0,5400,720
// ASSGN SYS003,DISK,VOL=SIPO31,SHR
// EXEC COPYSERV
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L228,CLASS=0,DISP=D
// JOB SP2L228 COMPARE PRIVATE OBJECTS WITH PRIVATE OBJECTS
* VSE/SP 2.1 LIBRARIAN ID=2 JOB=28
// EXEC LIBR
CONnect Sublib=SP2LIB.USRSLB1:=.USRSLB2
COMpare *.OBJ
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

COPYSERV	=====>	COM *.OBJ
UPSI -1	[=====>	CON S=SP2LIB.USRSLB1
IJSYSRL / SYSRLB	[	
UPSI -----1	[=====>	... :=.USRSLB2
IJSYSNR / SYS003	[	

```

* $$ JOB JNM=S31L229,CLASS=0,DISP=D
// JOB S31L229 COMPARE PRIVATE SOURCE WITH PRIVATE SOURCE
* SIPO 3.1 LIBRARIAN ID=2 JOB=29
* COPYSERV WILL ONLY RUN ON CKD DEVICES
// UPSI XX1XXX1X --1=PRIVATE SOURCE -----1=PRIVATE SOURCE
// DLBL IJSYSSL,'USER.SL.S31USL1.CKD',99/365,SD
// EXTENT SYSSLB,SIPO31,1,0,3240,720
// ASSGN SYSSLB,DISK,VOL=SIPO31,SHR
// DLBL IJSYSNR,'USER.SL.S31USL2.CKD',99/365,SD
// EXTENT SYS003,SIPO31,1,0,6120,720
// ASSGN SYS003,DISK,VOL=SIPO31,SHR
// EXEC COPYSERV
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L229,CLASS=0,DISP=D
// JOB SP2L229 COMAPRE PRIVATE SOURCE WITH PRIVATE SOURCE
* VSE/SP 2.1 LIBRARIAN ID=2 JOB=29
// EXEC LIBR
CONnect Sublib=SP2LIB.USRSLB1:=.USRSLB2
COMpare *.A,*.B,*.E
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

COPYSERV	=====>	COM *.A,*.B,*.E
UPSI --1	[=====>	CON S=SP2LIB.USRSLB1
IJSYSSL / SYSSLB	[	
UPSI -----1	[=====>	... :=.USRSLB2
IJSYSNR / SYS003	[	

```

* $$ JOB JNM=S31L230,CLASS=0,DISP=D
// JOB S31L230  BACKUP SYSTEM RESIDENCE - OLD METHOD
* SIPO 3.1 LIBRARIAN ID=2 JOB=30
// DLBL IJSYSRS,'PROD.CORE.IMAGE.LIBRARY.A',99/365,SD
// EXTENT SYS005,DOSRES,1,0,2,33950
// ASSGN SYS005,DISK,VOL=DOSRES,SHR
// ASSGN SYS006,1B0,D0
// MTC REW,SYS006
// EXEC BACKUP
  SA,NOSORT
/*
// MTC WTM,SYS006,5
// MTC REW,SYS006
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L230,CLASS=0,DISP=D
// JOB SP2L230  BACKUP IJSYSRS
* VSE/SP 2.1 LIBRARIAN ID=2 JOB=30
// ASSGN SYS006,1B3,D0
// EXEC LIBR
Backup Lib=IJSYSRS Tape=SYS006 Restore=Standalone
/*
// MTC WTM,SYS006,5
// MTC REW,SYS006
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

IJSYSRS / SYS005	=====>	L=IJSYSRS
SYS006	=====>	T=SYS006
BACKUP	=====>	B
SA,NOSORT	=====>	R=S

```

* $$ JOB JNM=S31L231,CLASS=0,DISP=D
// JOB S31L231  BACKUP PRIVATE LIBRARIES - NEW METHOD
* SIPO 3.1 LIBRARIAN  ID=2  JOB=31
// ASSGN SYS006,1B0,D0
// MTC REW,SYS006
// EXEC BACKUP
  NOSA,NOSORT,NODSF
  BACKUP CL,LIB=(S31UCL1,S31UCL2)
  BACKUP RL,LIB=(S31URL1,S31URL2)
  BACKUP SL,LIB=(S31USL1,S31USL2)
  BACKUP PL,LIB=(S31UPL1,S31UPL2)
/*
// MTC WTM,SYS006,5
// MTC REW,SYS006
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L231,CLASS=0,DISP=D
// JOB SP2L231  BACKUP SP2LIB PRIVATE USER LIBRARIES
* VSE/SP 2.1 LIBRARIAN  ID=2  JOB=31
// ASSGN SYS006,1B3,D0
// EXEC LIBR
Backup Lib=SP2LIB Tape=SYS006
/*
// MTC WTM,SYS006,5
// MTC REW,SYS006
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

SYS006	=====>	T=SYS006
BACKUP / NOSA,NOSORT, ...	=====>	B
BACKUP CL=(...)	[=====>	L=SP2LIB
BACKUP RL=(...)	[	
BACKUP SL=(...)	[	
BACKUP PL=(...)	[	

```

* $$ JOB JNM=S31L232,CLASS=0,DISP=D
// JOB S31L232 RESTORE SYSRES - OLD METHOD
* SIPO 3.1 LIBRARIAN ID=2 JOB=32
* RESTORE OF SYSRES TO NEW VOLUME
* RESTORE TO ORIGINAL VOLUME SHOULD BE DONE IN SA-MODE
// UPSI 111 ORIGINAL ALLOCATION VALUES
// DLBL IJSYSRS,'NEW.SYSRES.S31',99/365,SD
// EXTENT SYS005,SIPO31,1,0,2,35838
// ASSGN SYS005,DISK,VOL=SIPO31,SHR
// ASSGN SYS006,1B0,D0
// MTC REW,SYS006
// EXEC RESTORE
  NOVERIFY
/*
// MTC REW,SYS006
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L232,CLASS=0,DISP=D
// JOB SP2L232 RESTORE SYSRES
* VSE/SP 2.1 LIBRARIAN ID=2 JOB=32
* RESTORE OF SYSRES TO NEW VOLUME
* RESTORE TO ORIGINAL VOLUME MUST BE DONE IN SA-MODE
// ASSGN SYS006,1B3,D0
// MTC REW,SYS006
// EXEC LIBR
REStore Lib=IJSYSRS:IJSYSR1 Tape=SYS006 R=Y
/*
// MTC REW,SYS006
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

<u>SYS006</u>	<u>=====></u>	<u>T=SYS006</u>
<u>RESTORE / NOVERIFY</u>	<u>=====></u>	<u>RES</u>
<u>IJSYSRS / SYS005</u>	<u>=====></u>	<u>L=IJSYSRS:IJSYSR1</u>


```

* $$ JOB JNM=S31L233,CLASS=0,DISP=D
// JOB S31L233 RESTORE PRIVATE LIBRARIES - NEW METHOD
* SIPO 3.1 LIBRARIAN ID=2 JOB=33
// UPSI 0
// ASSGN SYS006,1B0,D0
// MTC REW,SYS006
// EXEC RESTORE
NOVERIFY
RESTORE CL,LIB=(S31UCL1,S31UCL2)
RESTORE RL,LIB=(S31URL1,S31URL2)
RESTORE SL,LIB=(S31USL1,S31USL2)
RESTORE PL,LIB=(S31UPL1,S31UPL2)
/*
// MTC REW,SYS006
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L233,CLASS=0,DISP=D
// JOB SP2L233 RESTORE PRIVATE LIBRARIES
* VSE/SP 2.1 LIBRARIAN ID=2 JOB=33
// ASSGN SYS006,1B3,D0
// MTC REW,SYS006
// EXEC LIBR
REStore Sublib=SP2LIB.USRSLB1,SP2LIB.USRSLB2 Tape=SYS006 R=Y
/*
// MTC REW,SYS006
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

SYS006	=====>	T=SYS006
RESTORE / NOVERIFY	=====>	RES
RESTORE CL,LIB=(...)	[=====>	S=SP2LIB.USRSLB1, ...
RESTORE RL,LIB=(...)	[	
RESTORE SL,LIB=(...)	[	
RESTORE PL,LIB=(...)	[	

```

* $$ JOB JNM=S31L234,CLASS=0,DISP=D
// JOB S31L234 LIST PRIVATE LIBRARIES DUMPED ON TAPE
* SIPO 3.1 LIBRARIAN ID=2 JOB=34
// UPSI 0
// ASSGN SYS006,1B0,D0
// MTC REW,SYS006
// EXEC RESTORE
  LIST DEVICE=FBA
/*
// MTC REW,SYS006
// EXEC RESTORE
  LIST DEVICE=3330
/*
// MTC REW,SYS006
// EXEC RESTORE
  LIST DEVICE=3340
/*
// MTC REW,SYS006
// EXEC RESTORE
  LIST DEVICE=3350
/*
// MTC REW,SYS006
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L234,CLASS=0,DISP=D
// JOB SP2L234 LIST PRIVATE LIBRARIES ON BACKUP TAPE
* VSE/SP 2.1 LIBRARIAN ID=2 JOB=34
// ASSGN SYS006,1B3,D0
// MTC REW,SYS006
// EXEC LIBR
  RESTORE * Tape=SYS006 SCan=Yes ID=*
/*
// MTC REW,SYS006
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

SYS006	=====>	T=SYS006
RESTORE	=====>	RES
LIST DEVICE=FBA	[=====>	* SC=Y ID=*
LIST DEVICE=3330	[	
LIST DEVICE=3340	[	
LIST DEVICE=3350	[	

Library Maintenance

```
* $$ JOB JNM=S31L301,CLASS=0,DISP=D
// JOB S31L301 CATALOG A PHASE WITH 'OPTION CATAL'
* SIPO 3.1 LIBRARIAN ID=3 JOB=01
// LIBDEF CL,TO=S31UCL2
// OPTION CATAL,NODECK
  PHASE SAMPLE,*
// EXEC ASSEMBLY
SAMPLE  START 0
          BALR  2,0
          USING *,2
          BR    14
          END   SAMPLE
/*
// EXEC LNKEDT
/*
/&
* $$ EOJ
```

```
* $$ JOB JNM=SP2L301,CLASS=0,DISP=D
// JOB SP2L301 CATALOG A PHASE WITH 'OPTION CATAL'
* VSE/SP 2.1 LIBRARIAN ID=3 JOB=01
// LIBDEF PHASE,CATALOG=SP2LIB.USRSLB2
// OPTION CATAL,NODECK
  PHASE SAMPLE,*
// EXEC ASSEMBLY
SAMPLE  START 0
          BALR  2,0
          USING *,2
          BR    14
          END   SAMPLE
/*
// EXEC LNKEDT
/*
/&
* $$ EOJ
```

MAPPING OF PARAMETERS

CL,TO=S31UCL2 =====> PHASE,CATALOG=SP2LIB. ...

```

* $$ JOB JNM=S31L302,CLASS=0,DISP=D
// JOB S31L302 CATAL AND EXEC A PHASE WITH 'OPTION LINK'
* SIPO 3.1 LIBRARIAN ID=3 JOB=02
// LIBDEF CL,TO=S31UCL2
// OPTION LINK,NODECK
// EXEC ASSEMBLY
SAMPLE START 0
        BALR 2,0
        USING *,2
        BR 14
        END SAMPLE

/*
// EXEC LNKEDT
// EXEC
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L302,CLASS=0,DISP=D
// JOB SP2L302 CATAL AND EXEC A PHASE WITH 'OPTION LINK'
* VSE/SP 2.1 LIBRARIAN ID=3 JOB=02
// OPTION LINK,NODECK
// EXEC ASSEMBLY
SAMPLE START 0
        BALR 2,0
        USING *,2
        SR 15,15 SET RETURN CODE TO ZERO
        BR 14
        END SAMPLE

/*
// EXEC LNKEDT
// EXEC
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

```

CL,TO=S31UCL2 =====> - - - - -

```

```

* $$ JOB JNM=S31L303,CLASS=0,DISP=D
// JOB S31L303 CATALOG A MODULE WITH 'CATALR'
* SIPO 3.1 LIBRARIAN ID=3 JOB=03
// LIBDEF RL,TO=S31URL2
// EXEC MAINT
   CATALR IJHAARCP
   ESD
   TXT
   TXT
   END
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L303,CLASS=0,DISP=D
// JOB SP2L303 CATALOG A MODULE IN USERLIB
* VSE/SP 2.1 LIBRARIAN ID=3 JOB=03
// EXEC LIBR
   Access Sublib=SP2LIB.USRSLB2
   Catalog IJHAARCP.OBJ Replace=Yes Eod=/+
   ESD
   TXT
   TXT
   END
/+
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

RL,TO=S31URL2	=====> A S=SP2LIB.USRSLB2
MAINT / CATALR IJHAARCP	=====> CA IJHAARCP.OBJ R=Y

```

* $$ JOB JNM=S31L304,CLASS=0,DISP=D
// JOB S31L304 CATALOG A SOURCE BOOK WITH 'CATALS'
* SIPO 3.1 LIBRARIAN ID=3 JOB=04
// LIBDEF SL,TO=S31USL2
// EXEC MAINT
    CATALS A.WAIT
    MACRO
        .
        .
        MEND
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L304,CLASS=0,DISP=D
// JOB SP2L304 CATALOG A SOURCE BOOK IN USERLIB
* VSE/SP 2.1 LIBRARIAN ID=3 JOB=04
// EXEC LIBR
    Access Sublib=SP2LIB.USRSLB2
    Catalog WAIT.A Replace=Yes Eod=/+
        .
        .
        .
/+
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

<u>SL,TO=S31USL2</u>	=====	> A S=SP2LIB.USRSLB2 E=/+
<u>MAINT / CATALS A.WAIT</u>	=====	> CA WAIT.A R=Y


```

* $$ JOB JNM=S31L305,CLASS=0,DISP=D
// JOB S31L305 CATALOG A PROCEDURE WITH 'CATALP'
* SIPO 3.1 LIBRARIAN ID=3 JOB=05
// LIBDEF PL,TO=S31UPL2
// EXEC MAINT
    CATALP LINKASM
// OPTION CATAL
    INCLUDE IPKLINK
// EXEC LNKEDT
    INCLUDE IPKESERV
// EXEC LNKEDT
/*
/+
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L305,CLASS=0,DISP=D
// JOB SP2L305 CATALOG A PROCEDURE
* VSE/SP 2.1 LIBRARIAN ID=3 JOB=05
// EXEC LIBR
Access Sublib=SP2LIB.USRSLB2
Catalog LINKASM.PROC Replace=Yes
// OPTION CATAL
    INCLUDE IPKLINK
// EXEC LNKEDT
    INCLUDE IPKESERV
// EXEC LNKEDT
/*
/+
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

PL,TO=S31UPL2	=====>	A S=SP2LIB.USRSLB2
MAINT / CATALP LINKASM	=====>	CA LINKASM.PROC R=Y

```

* $$ JOB JNM=S31L306,CLASS=0,DISP=D
// JOB S31L306  RENAME PHASES WITH 'RENAMC PN,PN,...'
* SIPO 3.1 LIBRARIAN  ID=3  JOB=06
// LIBDEF CL,TO=S31UCL2
// EXEC MAINT
   RENAMC DFHICP,DFHICOLD,DFHLFO,DFHLFOLD
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L306,CLASS=0,DISP=D
// JOB SP2L306  RENAME PHASES IN USERLIB
* VSE/SP 2.1 LIBRARIAN  ID=3  JOB=06
// EXEC LIBR
   Access Sublib=SP2LIB.USRSLB2
   RENAMC DFHICP.PHASE:DFHICOLD.=,DFHLFO.PHASE:DFHLFOLD.=
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

<u>CL,TO=S31LUCL2</u>	=====	> A S=SP2LIB.USRSLB2
<u>MAINT</u>	[=====	> REN DFHICP.PHASE:DFHICOLD.=
<u>RENAMC DFHICP,DFHICPLD</u>	[	

```

* $$ JOB JNM=S31L307,CLASS=0,DISP=D
// JOB S31L307  RENAME MODULES WITH 'RENAMR MN,MN,...'
* SIPO 3.1 LIBRARIAN  ID=3  JOB=07
// LIBDEF RL,TO=S31URL2
// EXEC MAINT
  RENAMR IJFFZZWZ,IJFFZOLD,IJDVZZIZ,IJDVZOLD
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L307,CLASS=0,DISP=D
// JOB SP2L307  RENAME OBJECTS IN USERLIB
* VSE/SP 2.1 LIBRARIAN  ID=3  JOB=07
// EXEC LIBR
  Access Sublib=SP2LIB.USRSLB2
  RENAME IJFFZZWZ.OBJ:IJFFZOLD.=,IJDVZZIZ.OBJ:IJDVZOLD.=
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

RL,TO=S31URL2	=====	> A S=SP2LIB.USRSLB2
MAINT	[=====	> REN IJFFZZWZ.OBJ:IJFFZOLD.=
RENAMR IJFFSSWZ,IJFFZOLD	[	

```

* $$ JOB JNM=S31L308,CLASS=0,DISP=D
// JOB S31L308  RENAME SOURCE BOOKS WITH 'RENAMS S.B,S.B,...'
* SIPO 3.1 LIBRARIAN  ID=3  JOB=08
// LIBDEF SL,TO=S31USL2
// EXEC MAINT
  RENAMS B.VTAM,B.VTAMOLD,E.DEQ,E.DEQOLD
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L308,CLASS=0,DISP=D
// JOB SP2L308  RENAME SOURCE BOOKS IN USERLIB
* VSE/SP 2.1 LIBRARIAN  ID=3  JOB=08
// EXEC LIBR
  Access Sublib=SP2LIB.USRSLB2
  RENAME VTAM.B:VTAMOLD.B,DEQ.E:DEQOLD.E
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

SL,TO=S31USL2	=====> A S=SP2LIB.USRSLB2
MAINT	[=====> REN VTAM.B:VTAMOLD.B
RENAMS B.VTAM,B.VTAMOLD	[

```

* $$ JOB JNM=S31L309,CLASS=0,DISP=D
// JOB S31L309  RENAME PROCEDURES WITH 'RENAMP PN,PN,...'
* SIPO 3.1 LIBRARIAN  ID=3  JOB=09
// LIBDEF PL,TO=S31UPL2
// EXEC MAINT
  RENAMP LABWK21,LBOLD,CONDENSE,CONDOLD
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L309,CLASS=0,DISP=D
// JOB SP2L309  RENAME PROCS IN USERLIB
* VSE/SP 2.1 LIBRARIAN  ID=3  JOB=09
// EXEC LIBR
Access Sublib=SP2LIB.USRSLB2
REName LABWK21.PROC:LBOLD.=,CONDENSE.PROC:CONDOLD.=
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

<u>PL,TO=S31UPL2</u>	=====> A S=SP2LIB.USRSLB2
<u>MAINT</u>	[=====> REN LABWK21.PROC:LBOLD.=
<u>RENAMP LABWK21,LBOLD</u>	

```

* $$ JOB JNM=S31L310,CLASS=0,DISP=D
// JOB S31L309  UPDATE MEMBER OF SOURCE STATEMENT LIBRARY
* SIPO 3.1 LIBRARIAN  ID=3  JOB=10
// LIBDEF SL,TO=S31USL2
// EXEC MAINT
UPDATE A.RJEVERF,,,10
) REP 0040
    B. TAILOR VSE/POWER VERSION 2
) DEL 0860,0870
) ADD 1080
HOPEFULLY THIS UPDATE JOB WAS RUNNING SUCCESSFULLY.
) END
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L310,CLASS=0,DISP=D
// JOB SP2L310  UPDATE SOURCE BOOK IN USERLIB
* VSE/SP 2.1 LIBRARIAN  ID=3  JOB=10
// EXEC LIBR
Access Sublib=SP2LIB.USRSLB2
Update RJEVERF.A SEquence=10
) REP 0040
    B. TAILOR VSE/POWER VERSION 2
) DEL 0860,0870
) ADD 1080
HOPEFULLY THIS UPDATE JOB WAS RUNNING SUCCESSFULLY.
) END
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

<u>SL,TO=S31USL2</u>	=====>	<u>A S=SP2LIB.USRSLB2</u>
<u>MAINT / UPDATE A.RJE...</u>	=====>	<u>U RJEVERF.A SE=10</u>

```

* $$ JOB JNM=S31L311,CLASS=0,DISP=D
// JOB S31L311 DELETE PHASES WITH 'DELETC PN,PN,...'
* SIPO 3.1 LIBRARIAN ID=3 JOB=11
// LIBDEF CL,TO=S31UCL2
// EXEC MAINT
DELETC COMPHIST,ACCPASS
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L311,CLASS=0,DISP=D
// JOB SP2L311 DELETE SELECTED PHASES FROM USERLIB
* VSE/SP 2.1 LIBRARIAN ID=3 JOB=11
// EXEC LIBR
Access Sublib=SP2LIB.USRSLB2
DELeTe COMPHIST.PHASE,ACCPASS.PHASE
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

CL,TO=S31UCL2	=====> A S=SP2LIB.USRSLB2
MAINT	[=====> DEL COMPHIST.PHASE
DELETC COMPHIST	[

```

* $$ JOB JNM=S31L312,CLASS=0,DISP=D
// JOB S31L312 DELETE PHASES WITH 'DELETC XXXX.ALL'
* SIPO 3.1 LIBRARIAN ID=3 JOB=12
// LIBDEF CL,TO=S31UCL2
// EXEC MAINT
  DELETC DFHT.ALL
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L312,CLASS=0,DISP=D
// JOB SP2L312 DELETE ALL DFHT PHASES FROM USER LIB
* VSE/SP 2.1 LIBRARIAN ID=3 JOB=12
// EXEC LIBR
Access Sublib=SP2LIB.USRSLB2
DELeTe DFHT*.PHASE
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

CL,TO=S31UCL2	=====> A S=SP2LIB.USRSLB2
MAINT	[=====> DEL DFHT*.PHASE
DELETC DFHT.ALL	[


```

* $$ JOB JNM=S31L313,CLASS=0,DISP=D
// JOB S31L313 DELETE PHASES WITH 'DELETC ALL'
* SIPO 3.1 LIBRARIAN ID=3 JOB=13
// LIBDEF CL,TO=S31UCL2
// EXEC MAINT
  DELETC ALL
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L313,CLASS=0,DISP=D
// JOB SP2L313 DELETE ALL PHASES FROM USERLIB 2
* VSE/SP 2.1 LIBRARIAN ID=3 JOB=13
// EXEC LIBR
Access Sublib=SP2LIB.USRSLB2
DELEte *.PHASE
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

CL,TO=S31UCL2	=====> A S=SP2LIB.USRSLB2
MAINT	[=====> DEL *.PHASE
DELETC ALL	[

```

* $$ JOB JNM=S31L314,CLASS=0,DISP=D
// JOB S31L314 DELETE MODULES WITH 'DELETR MN,MN,...'
* SIPO 3.1 LIBRARIAN ID=3 JOB=14
// LIBDEF RL,TO=S31URL2
// EXEC MAINT
DELETR IJB116CF,IJBTVYN
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L314,CLASS=0,DISP=D
// JOB SP2L314 DELETE SELECTED OBJECT MODULES
* VSE/SP 2.1 LIBRARIAN ID=3 JOB=14
// EXEC LIBR
Access Sublib=SP2LIB.USRSLB2
DELeTe IJB116CF.OBJ,IJBTVYN.OBJ
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

<u>RL,TO=S31URL2</u>	<u>=====> A S=SP2LIB.USRSLB2</u>
<u>MAINT</u>	<u>[=====> DEL IJB116CF.OBJ</u>
<u>DELETR IJB116CF</u>	<u>[</u>

```

* $$ JOB JNM=S31L315,CLASS=0,DISP=D
// JOB S31L315  DELETE MODULES WITH 'DELETR XXX.ALL'
* SIPO 3.1 LIBRARIAN  ID=3  JOB=15
// LIBDEF RL,TO=S31URL2
// EXEC MAINT
  DELETR IJD.ALL
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L315,CLASS=0,DISP=D
// JOB SP2L315  DELETE ALL IJD OBJECT MODULES
* VSE/SP 2.1 LIBRARIAN  ID=3  JOB=15
// EXEC LIBR
  Access Sublib=SP2LIB.USRSLB2
  DELeTe IJD*.OBJ
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

RL,TO=S31URL2	=====	> A S=SP2LIB.USRSLB2
MAINT	=====	> DEL IJD*.OBJ
DELETR IJD.ALL		

```

* $$ JOB JNM=S31L316,CLASS=0,DISP=D
// JOB S31L316 DELETE MODULES WITH 'DELETR ALL'
* SIPO 3.1 LIBRARIAN ID=3 JOB=16
// LIBDEF RL,TO=S31URL2
// EXEC MAINT
  DELETR ALL
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L316,CLASS=0,DISP=D
// JOB SP2L316 DELETE ALL OBJECT MODULES FROM USERLIB
* VSE/SP 2.1 LIBRARIAN ID=3 JOB=16
// EXEC LIBR
Access Sublib=SP2LIB.USRSLB2
DELeTe *.OBJ
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

RL,TO=S31URL2	=====> A S=SP2LIB.USRSLB2
MAINT	[=====> DEL *.OBJ
DELETR ALL	[

```

* $$ JOB JNM=S31L317,CLASS=0,DISP=D
// JOB S31L317 DELETE SOURCE BOOKS WITH 'DELETS S.B,S.B,...'
* SIPO 3.1 LIBRARIAN ID=3 JOB=17
// LIBDEF SL,TO=S31USL2
// EXEC MAINT
DELETS A.DDPUSSTB,E.WAIT
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L317,CLASS=0,DISP=D
// JOB SP2L317 DELETE SELECTED SOURCE BOOKS
* VSE/SP 2.1 LIBRARIAN ID=3 JOB=17
// EXEC LIBR
Access Sublib=SP2LIB.USRSLB2
DELeTe DDPUSSTB.A,WAIT.E
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

SL,TO=S31USL2	=====> A S=SP2LIB.USRSLB2
MAINT	[=====> DEL DDPUSSTB.A
DELETS A.DDPUSSTB	[

```

* $$ JOB JNM=S31L318,CLASS=0,DISP=D
// JOB S31L318  DELETE SOURCE BOOKS WITH 'DELETS X.ALL'
* SIPO 3.1 LIBRARIAN  ID=3  JOB=18
// LIBDEF SL,TO=S31USL2
// EXEC MAINT
  DELETS B.ALL
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L318,CLASS=0,DISP=D
// JOB SP2L318  DELETE ALL B. SOURCE BOOKS
* VSE/SP 2.1 LIBRARIAN  ID=3  JOB=18
// EXEC LIBR
Access Sublib=SP2LIB.USRSLB2
DElete *.B
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

SL,TO=S31USL2	=====> A S=SP2LIB.USRSLB2
MAINT	[=====> DEL *.B
DELETS B.ALL	[

```

* $$ JOB JNM=S31L319,CLASS=0,DISP=D
// JOB S31L319 DELETE SOURCE BOOKS WITH 'DELETS ALL'
* SIPO 3.1 LIBRARIAN ID=3 JOB=19
// LIBDEF SL,TO=S31USL2
// EXEC MAINT
  DELETS ALL
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L319,CLASS=0,DISP=D
// JOB SP2L319 DELETE ALL SOURCE BOOKS FROM USERLIB
* VSE/SP 2.1 LIBRARIAN ID=3 JOB=19
// EXEC LIBR
Access Sublib=SP2LIB.USRSLB2
DELeTe *.A,*.B,*.E
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

SL,TO=S31USL2	=====> A S=SP2LIB.USRSLB2
MAINT	[=====> DEL *.A,*.B,*.E
DELETS ALL	[

```

* $$ JOB JNM=S31L320,CLASS=0,DISP=D
// JOB S31L320  DELETE PROCEDURES WITH 'DELETP PN,PN,...'
* SIPO 3.1 LIBRARIAN  ID=3  JOB=20
// LIBDEF PL,TO=S31UPL2
// EXEC MAINT
DELETP LABWK33,LABWK44
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L320,CLASS=0,DISP=D
// JOB SP2L320  DELETE SELECTED PROCS
* VSE/SP 2.1 LIBRARIAN  ID=3  JOB=20
// EXEC LIBR
Access Sublib=SP2LIB.USRSLB2
DELeTe LABWK33.PROC,LABWK44.PROC
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

PL,TO=S31UPL2	=====> A S=SP2LIB.USRSLB2
MAINT	[=====> DEL LABWK33.PROC
DELETP LABWK33	[


```

* $$ JOB JNM=S31L321,CLASS=0,DISP=D
// JOB S31L321 DELETE PROCEDURES WITH 'DELETP ALL'
* SIPO 3.1 LIBRARIAN ID=3 JOB=21
// LIBDEF PL,TO=S31UPL2
// EXEC MAINT
  DELETP ALL
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L321,CLASS=0,DISP=D
// JOB SP2L321 DELETE ALL PROCS FROM USERLIB
* VSE/SP 2.1 LIBRARIAN ID=3 JOB=21
// EXEC LIBR
Access Sublib=SP2LIB.USRSLB2
DELEte *.PROC
/*
/&
* $$ EOJ

```

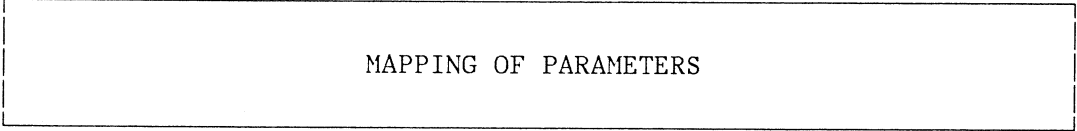
MAPPING OF PARAMETERS

PL,TO=S31UPL2	=====> A S=SP2LIB.USRSLB2
MAINT	[=====> DEL *.PROC
DELETP ALL	[

Library Services

```
* $$ JOB JNM=S31L401,CLASS=0,DISP=D
// JOB S31L401 DSERV STATUS OF PRIVATE LIBRARIES
* SIPO 3.1 LIBRARIAN ID=4 JOB=01
// LIBDEF CL,FROM=S31UCL2
// LIBDEF RL,FROM=S31URL2
// LIBDEF SL,FROM=S31USL2
// LIBDEF PL,FROM=S31UPL2
// EXEC DSERV
/*
/&
* $$ EOJ
```

```
* $$ JOB JNM=SP2L401,CLASS=0,DISP=D
// JOB SP2L401 DISPLAY STATUS OF USERLIB
* VSE/SP 2.1 LIBRARIAN ID=4 JOB=01
// EXEC LIBR
LISTD Sublib=SP2LIB.USRSLB2 Output=STatus Unit=SYSLST
/*
/&
* $$ EOJ
```



CL,FROM=S31UCL2	[=====>	A S=SP2LIB.USRSLB2
RL,FROM=S31URL2	[	
SL,FROM=S31USL2	[	
PL,FROM=S31UPL2	[	
DSERV	=====>	LISTD O=ST

*** \$\$ JOB JNM=S31L402,CLASS=0,DISP=D**

// JOB S31L402 DSERV ALL LIBRARIES

* SIPO 3.1 LIBRARIAN ID=4 JOB=02

// LIBDEF CL,FROM=S31UCL2

// LIBDEF RL,FROM=S31URL2

// LIBDEF SL,FROM=S31USL2

// LIBDEF PL,FROM=S31UPL2

// EXEC DSERV

DSPLYS ALL

/*

/&

* \$\$ EOJ

*** \$\$ JOB JNM=SP2L402,CLASS=0,DISP=D**

// JOB SP2L402 DISPLAY DIRECTORY OF USERLIB

* VSE/SP 2.1 LIBRARIAN ID=4 JOB=02

// EXEC LIBR

LISTD Sublib=SP2LIB.USRSLB2 Output=Normal Unit=SYSLST

/*

/&

* \$\$ EOJ

MAPPING OF PARAMETERS

CL,FROM=S31UCL2	[=====>	A S=SP2LIB.USRSLB2
RL,FROM=S31URL2	[	
SL,FROM=S31USL2	[	
PL,FROM=S31UPL2	[	
DSERV / DSPLYS ALL	=====>	LISTD O=ST

```

* $$ JOB JNM=S31L403,CLASS=0,DISP=D
// JOB S31L403 DSERV SYSTEM DIRECTORY LIST
* SIPO 3.1 LIBRARIAN ID=4 JOB=03
// LIBDEF CL,FROM=S31UCL2
// EXEC DSERV
  DSDPLY SDL
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L403,CLASS=0,DISP=D
// JOB SP2L403 DISPLAY SYSTEM DIRECTORY LIST
* VSE/SP 2.1 LIBRARIAN ID=4 JOB=03
// EXEC LIBR
  LISTD SDL
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

CL,FROM=S31UCL2	=====>	- - - - -
DSERV/ DSDPLY SDL	=====>	LISTD SDL

```

* $$ JOB JNM=S31L404,CLASS=0,DISP=D
// JOB S31L404 DSERV TRANSIENT DIRECTORY
* SIPO 3.1 LIBRARIAN ID=4 JOB=04
// LIBDEF CL,FROM=S31UCL2
// EXEC DSERV
  DSPLY TD
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L404,CLASS=0,DISP=D
// JOB SP2L404 DSERV ALL TRANSIENTS - ALSO IN 'LD SDL'
* VSE/SP 2.1 LIBRARIAN ID=4 JOB=04
// EXEC LIBR
Access Sublib=SP2LIB.USRSLB2
LISTD $$*.PHASE Output=Normal Unit=SYSLST
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

CL,FROM=S31UCL2	=====>	A S=SP2LIB.USRSLB2
DSERV / DSPLY TD	=====>	LISTD \$\$*.PHASE

```

* $$ JOB JNM=S31L405,CLASS=0,DISP=D
// JOB S31L405 DSERV CORE IMAGE DIRECTORY
* SIPO 3.1 LIBRARIAN ID=4 JOB=05
// LIBDEF CL,FROM=S31UCL2
// EXEC DSERV
  DSPLY CD
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L405,CLASS=0,DISP=D
// JOB SP2L405 DSERV ALL 'CORE IMAGE' PHASES
* VSE/SP 2.1 LIBRARIAN ID=4 JOB=05
// EXEC LIBR
Access Sublib=SP2LIB.USRSLB2
LISTD *.PHASE Output=Normal Unit=SYSLST
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

CL,FROM=S31UCL2	=====>	A S=SP2LIB.USRSLB2
DSERV / DSPLY CD	=====>	LISTD *.PHASE

```

* $$ JOB JNM=S31L406,CLASS=0,DISP=D
// JOB S31L406 DSERV SELECTED PHASES OF CORE IMAGE DIRECTORY
* SIPO 3.1 LIBRARIAN ID=4 JOB=06
// LIBDEF CL,FROM=S31UCL2
// EXEC DSERV
  D$PLY CD(ISTI)
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L406,CLASS=0,DISP=D
// JOB SP2L406 DSERV SELECTED PHASES
* VSE/SP 2.1 LIBRARIAN ID=4 JOB=06
// EXEC LIBR
Access Sublib=SP2LIB.USRSLB2
LISTD ISTI*.PHASE Output=Normal Unit=SYSLST
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

<u>CL, FROM=S31UCL2</u>	=====>	<u>A S=SP2LIB.USRSLB2</u>
<u>DSERV / D\$PLY CD(ISTI)</u>	=====>	<u>LISTD ISTI*.PHASE</u>

```

* $$ JOB JNM=S31L407,CLASS=0,DISP=D
// JOB S31L407 DSERV RELOCATABLE DIRECTORY
* SIPO 3.1 LIBRARIAN ID=4 JOB=07
// LIBDEF RL,FROM=S31URL2
// EXEC DSERV
  DSPLYS RD
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L407,CLASS=0,DISP=D
// JOB SP2L407 DSERV ALL RELOCATABLE OBJECTS
* VSE/SP 2.1 LIBRARIAN ID=4 JOB=07
// EXEC LIBR
Access Sublib=SP2LIB.USRSLB2
LISTD *.OBJ Output=Normal Unit=SYSLST
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

RL, FROM=S31URL2	=====>	A S=SP2LIB.USRSLB2
DSERV / DSPLYS RD	=====>	LISTD *.OBJ


```

* $$ JOB JNM=S31L408,CLASS=0,DISP=D
// JOB S31L408 DSERV SOURCE STATEMENT DIRECTORY
* SIPO 3.1 LIBRARIAN ID=4 JOB=08
// LIBDEF SL,FROM=S31USL2
// EXEC DSERV
  DSPLYS SD
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L408,CLASS=0,DISP=D
// JOB SP2L408 DSERV ALL SOURCE BOOKS
* VSE/SP 2.1 LIBRARIAN ID=4 JOB=08
// EXEC LIBR
Access Sublib=SP2LIB.USRSLB2
LISTD *.A,*.B,*.E Output=Normal Unit=SYSLST
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

SL,FROM=S31USL2	=====>	A S=SP2LIB.USRSLB2
DSERV / DSPLYS SD	=====>	LISTD *.A,*.B,*.E

```

* $$ JOB JNM=S31L409,CLASS=0,DISP=D
// JOB S31L409 DSERV PROCEDURE DIRECTORY
* SIPO 3.1 LIBRARIAN ID=4 JOB=09
// LIBDEF PL,FROM=S31UPL2
// EXEC DSERV
  DSPLYS PD
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L409,CLASS=0,DISP=D
// JOB SP2L409 DSERV ALL PROCEDURES
* VSE/SP 2.1 LIBRARIAN ID=4 JOB=09
// EXEC LIBR
Access Sublib=SP2LIB.USRSLB2
LISTD *.PROC Output=Normal Unit=SYSLST
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

PL,FROM=S31UPL2	=====>	A S=SP2LIB.USRSLB2
DSERV / DSPLYS PD	=====>	LISTD *.PROC

```

* $$ JOB JNM=S31L410,CLASS=0,DISP=D
// JOB S31L410  DISPLAY ALL PHASES FROM CIL
* SIPO 3.1 LIBRARIAN  ID=4  JOB=10
// LIBDEF CL,FROM=S31UCL2
// EXEC CSERV
  DSPLY ALL
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L410,CLASS=0,DISP=D
// JOB SP2L410  LIST ALL PHASES FROM USERLIB
* VSE/SP 2.1 LIBRARIAN  ID=4  JOB=10
// EXEC LIBR
Access Sublib=SP2LIB.USRSLB2
List *.PHASE Unit=SYSLST
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

CL, FROM=S31UCL2	=====> A S=SP2LIB.USRSLB2
CSERV / DSPLY ALL	=====> L *.PHASE

```

* $$ JOB JNM=S31L411,CLASS=0,DISP=D
* $$ PUN CLASS=Q,DISP=L
// JOB S31L411 PUNCH SELECTED PHASES FROM CIL
* SIPO 3.1 LIBRARIAN ID=4 JOB=11
// LIBDEF CL,FROM=S31UCL2
// EXEC CSERV
PUNCH DFHI.ALL
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L411,CLASS=0,DISP=D
* $$ PUN CLASS=Q,DISP=L
// JOB SP2L411 PUNCH SELECTED PHASES FROM USERLIB
* VSE/SP 2.1 LIBRARIAN ID=4 JOB=11
// EXEC LIBR
Access Sublib=SP2LIB.USRSLB2
PUnch DFHI*.PHASE
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

<u>CL,FROM=S31UCL2</u>	=====>	<u>A S=SP2LIB.USRSLB2</u>
<u>CSERV / PUNCH DFHI.ALL</u>	=====>	<u>P DFHI*.PHASE</u>

```

* $$ JOB JNM=S31L412,CLASS=0,DISP=D
* $$ PUN CLASS=Q,DISP=L
// JOB S31L412  DISPLAY AND PUNCH SELECTED PHASES FROM CIL
* SIPO 3.1 LIBRARIAN  ID=4  JOB=12
// LIBDEF CL,FROM=S31UCL2
// EXEC CSERV
   DSPCH DFHPLP,DFHRTE
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L412,CLASS=0,DISP=D
* $$ PUN CLASS=Q,DISP=L
// JOB SP2L412  DISPLAY AND PUNCH SELECTED PHASES FROM USERLIB
* VSE/SP 2.1 LIBRARIAN  ID=4  JOB=12
// EXEC LIBR
Access Sublib=SP2LIB.USRSLB2
List DFHPLP.PHASE,DFHRTE.PHASE Unit=SYSLST
PUnch DFHPLP.PHASE,DFHRTE.PHASE
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

CL,FROM=S31UCL2	=====	> A S=SP2LIB.USRSLB2
CSERV / DSPCH DFHPLP	=====	]> L DFHPLP.PHASE
		]> PU DFHPLP.PHASE

```

* $$ JOB JNM=S31L413,CLASS=0,DISP=D
// JOB S31L413  DISPLAY MODULES FROM RELOCATABLE LIBRARY
* SIPO 3.1 LIBRARIAN  ID=4  JOB=13
// LIBDEF RL,FROM=S31URL2
// EXEC RSERV
DSPLY ALL
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L413,CLASS=0,DISP=D
// JOB SP2L413  DISPLAY RELOCATABLE MODULES FROM USERLIB
* VSE/SP 2.1 LIBRARIAN  ID=4  JOB=13
// EXEC LIBR
Access Sublib=SP2LIB.USRSLB2
List *.OBJ Unit=SYSLST
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

<u>RL,FROM=S31URL2</u>	=====>	<u>A S=SP2LIB.USRSLB2</u>
<u>RSERV / DSPLY ALL</u>	=====>	<u>L *.OBJ</u>

```

* $$ JOB JNM=S31L414,CLASS=0,DISP=D
* $$ PUN CLASS=Q,DISP=L
// JOB S31L414 PUNCH SELECTED MODULES FROM RELOCATABLE LIBRARY
* SIPO 3.1 LIBRARIAN ID=4 JOB=14
// LIBDEF RL,FROM=S31URL2
// EXEC RSERV
PUNCH IJG.ALL
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L414,CLASS=0,DISP=D
* $$ PUN CLASS=Q,DISP=L
// JOB SP2L414 PUNCH SELECTED RELOCATABLE OBJECTS
* VSE/SP 2.1 LIBRARIAN ID=4 JOB=14
// EXEC LIBR
Access Sublib=SP2LIB.USRSLB2
PUnch IJG*.OBJ
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

RL,FROM=S31URL2	=====>	A S=SP2LIB.USRSLB2
RESERV / PUNCH IJG.ALL	=====>	PU IJG*.OBJ

```

* $$ JOB JNM=S31L415,CLASS=0,DISP=D
* $$ PUN CLASS=Q,DISP=L
// JOB S31L415  DISPLAY AND PUNCH MODULES FROM RELOC LIBRARY
* SIPO 3.1 LIBRARIAN  ID=4  JOB=15
// LIBDEF RL,FROM=S31URL2
// EXEC RSERV
  DSPCH IJHAARCP,IKQSMSSF
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L415,CLASS=0,DISP=D
* $$ PUN CLASS=Q,DISP=L
// JOB SP2L415  DISPLAY AND PUNCH SELECTED OBJECTS FROM USERLIB
* VSE/SP 2.1 LIBRARIAN  ID=4  JOB=15
// EXEC LIBR
Access Sublib=SP2LIB.USRSLB2
List IJHAARCP.OBJ,IKQSMSSF.OBJ Unit=SYSLST
PUnch IJHAARCP.OBJ,IKQSMSSF.OBJ
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

<u>RL,FROM=S31URL2</u>	=====>	<u>A S=SP2LIB.USRSLB2</u>
<u>RESERV / DSPCH IJHAARCP</u>	===== >	<u>L IJHAARCP.OBJ</u>
	>	<u>PU IJHAARCP.OBJ</u>


```

* $$ JOB JNM=S31L416,CLASS=0,DISP=D
// JOB S31L416 DISPLAY SOURCE STATEMENT LIBRARY
* SIPO 3.1 LIBRARIAN ID=4 JOB=16
// LIBDEF SL,FROM=S31USL2
// EXEC SSERV
  DSPLY ALL
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L416,CLASS=0,DISP=D
// JOB SP2L416 DISPLAY ALL SOURCE BOOKS
* VSE/SP 2.1 LIBRARIAN ID=4 JOB=16
// EXEC LIBR
Access Sublib=SP2LIB.USRSLB2
List *.A,*.B,*.E Unit=SYSLST
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

<u>SL,FROM=S31USL2</u>	=====>	A S=SP2LIB.USRSLB2
<u>SSERV / DSPLY ALL</u>	=====>	L *.A,*.B,*.E

```

* $$ JOB JNM=S31L417,CLASS=0,DISP=D
// JOB S31L417  DISPLAY MEMBERS FROM SOURCE STATEMENT LIBRARY IN HEX
* SIPO 3.1 LIBRARIAN  ID=4  JOB=17
// LIBDEF SL,FROM=S31USL2
// EXEC SSERV
  DSPLY E.PFREE,E.PFIX,HEX
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L417,CLASS=0,DISP=D
// JOB SP2L417  DISPLAY SELECTED SOURCE BOOKS IN HEX
* VSE/SP 2.1 LIBRARIAN  ID=4  JOB=17
// EXEC LIBR
Access Sublib=SP2LIB.USRSLB2
List PFREE.E,PFIX.E Unit=SYSLST Format=Hex
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

SL,FROM=S31USL2	=====>	A S=SP2LIB.USRSLB2
SSERV / DSPLY E.PFREE	=====>	L PFREE.E
HEX	=====>	F=H

```

* $$ JOB JNM=S31L418,CLASS=0,DISP=D
* $$ PUN CLASS=Q,DISP=L
// JOB S31L418 PUNCH SELECTED MEMBERS FROM SSL
* SIPO 3.1 LIBRARIAN ID=4 JOB=18
// LIBDEF SL,FROM=S31USL2
// EXEC SSERV
  PUNCH B.ALL
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L418,CLASS=0,DISP=D
* $$ PUN CLASS=Q,DISP=L
// JOB SP2L418 PUNCH SELECTED SOURCE BOOKS
* VSE/SP 2.1 LIBRARIAN ID=4 JOB=18
// EXEC LIBR
Access Sublib=SP2LIB.USRSLB2
PUnch *.B
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

SL,FROM=S31USL2	=====>	A S=SP2LIB.USRSLB2
SSERV / PUNCH B.ALL	=====>	PU *.B

```

* $$ JOB JNM=S31L419,CLASS=0,DISP=D
* $$ PUN CLASS=Q,DISP=L
// JOB S31L419 DISPLAY AND PUNCH SELECTED MEMBERS FROM SSL
* SIPO 3.1 LIBRARIAN ID=4 JOB=19
// LIBDEF SL,FROM=S31USL2
// EXEC SSERV
  DSPCH A.RJEVERF,B.APP
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L419,CLASS=0,DISP=D
* $$ PUN CLASS=Q,DISP=L
// JOB SP2L419 DISPLAY AND PUNCH SELECTED SOURCE BOOKS
* VSE/SP 2.1 LIBRARIAN ID=4 JOB=19
// EXEC LIBR
Access Sublib=SP2LIB.USRSLB2
List RJEVERF.A,APP.B Unit=SYSLST
PUnch RJEVERF.A,APP.B
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

SL,FROM=S31USL2	=====>	A S=SP2LIB.USRSLB2
SSERV / DSPCH A.RJEVERF	=====]>	L RJEVERF.A
	]>	PU RJEVERF.A

```

* $$ JOB JNM=S31L420,CLASS=0,DISP=D
// JOB S31L420  DISPLAY PROCEDURE LIBRARY
* SIPO 3.1 LIBRARIAN  ID=4  JOB=20
// LIBDEF PL,FROM=S31UPL2
// EXEC PSERV
  DSPLY ALL
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L420,CLASS=0,DISP=D
// JOB SP2L420  DISPLAY PROCEDURES IN USERLIB
* VSE/SP 2.1 LIBRARIAN  ID=4  JOB=20
// EXEC LIBR
Access Sublib=SP2LIB.USRSLB2
List *.PROC Unit=SYSLST
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

PL,FROM=S31UPL2	=====>	A S=SP2LIB.USRSLB2
PSERV / DSPLY ALL	=====>	L *.PROC

```

* $$ JOB JNM=S31L421,CLASS=0,DISP=D
* $$ PUN CLASS=Q,DISP=L
// JOB S31L421 PUNCH SELECTED MEMBERS FROM PROCEDURE LIBRARY
* SIPO 3.1 LIBRARIAN ID=4 JOB=21
// LIBDEF PL,FROM=S31UPL2
// EXEC PSERV
PUNCH LINKASM,IPLSUP2
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L421,CLASS=0,DISP=D
* $$ PUN CLASS=Q,DISP=L
// JOB SP2L421 PUNCH SELECTED PROCEDURES FROM USERLIB
* VSE/SP 2.1 LIBRARIAN ID=4 JOB=21
// EXEC LIBR
Access Sublib=SP2LIB.USRSLB2
PUnch LINKASM.PROC,IPLSUP2.PROC
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

<u>PL,FROM=S31UPL2</u>	=====>	<u>A S=SP2LIB.USRSLB2</u>
<u>PSERV / PUNCH LINKASM</u>	=====>	<u>PU LINKASM.PROC</u>

```

* $$ JOB JNM=S31L422,CLASS=0,DISP=D
* $$ PUN CLASS=Q,DISP=L
// JOB S31L422 DISPLAY AND PUNCH SELECTED MEMBERS FROM PROCLIB
* SIPO 3.1 LIBRARIAN ID=4 JOB=22
// LIBDEF PL,FROM=S31UPL2
// EXEC PSERV
  DSPCH FASTFTCH,LINKDSF
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L422,CLASS=0,DISP=D
* $$ PUN CLASS=Q,DISP=L
// JOB SP2L422 DISPLAY AND PUNCH SELECTED PROCEDURES
* VSE/SP 2.1 LIBRARIAN ID=4 JOB=22
// EXEC LIBR
Access Sublib=SP2LIB.USRSLB2
List FASTFTCH.PROC,LINKDSF.PROC Unit=SYSLST
Punch FASTFTCH.PROC,LINKDSF.PROC
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

PL,FROM=S31UPL2	=====>	A S=SP2LIB.USRSLB2
PSERV / DSPCH FASTFTCH	=====]>	L FASTFTCH.PROC
	]>	PU FASTFTCH.PROC

E-MACRO Maintenance and Services

```
* $$ JOB JNM=S31L501,CLASS=0,DISP=D
* $$ PUN CLASS=Q,DISP=L
// JOB S31L501  DISPLAY AND PUNCH E.WAIT
* SIPO 3.1 LIBRARIAN  ID=5  JOB=01
// LIBDEF SL,FROM=S31USL2
// EXEC ESERV
  DSPCH E.WAIT
/*
/&
* $$ EOJ
```

```
* $$ JOB JNM=SP2L501,CLASS=0,DISP=D
* $$ PUN CLASS=Q,DISP=L
// JOB SP2L501  DISPLAY AND PUNCH E.WAIT
* VSE/SP 2.1 LIBRARIAN  ID=5  JOB=01
// LIBDEF SOURCE,SEARCH=SP2LIB.USRSLB2
// EXEC ESERV
  DSPCH E.WAIT
/*
/&
* $$ EOJ
```

MAPPING OF PARAMETERS

SL,FROM=S31USL2 =====> SOURCE,SEARCH=SP2LIB. ...


```

* $$ JOB JNM=S31L502,CLASS=0,DISP=D
* $$ PUN CLASS=Q,DISP=L
// JOB S31L502 PUNCH AND UPDATE E.WAIT
* SIPO 3.1 LIBRARIAN ID=5 JOB=02
// LIBDEF SL,FROM=S31USL2
// EXEC ESERV
  PUNCH E.WAIT
) COL 77,4
) VER 0028,50
* SUPERVISOR - WAIT - 5745-SC-SUP - REL. 28.0
) REP 0028
* SUPERVISOR - WAIT - 5745-SC-SUP - VSE REL. 3
) END
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L502,CLASS=0,DISP=D
* $$ PUN CLASS=Q,DISP=L
// JOB SP2L502 PUNCH AND UPDATE E.WAIT
* VSE/SP 2.1 LIBRARIAN ID=5 JOB=02
// LIBDEF SOURCE,SEARCH=SP2LIB.USRSLB2
// EXEC ESERV
  PUNCH E.WAIT
) COL 77,4
) VER 0028,50
* SUPERVISOR - WAIT - 5745-SC-SUP - REL. 28.0
) REP 0028
* SUPERVISOR - WAIT - 5745-SC-SUP - VSE REL. 3
) END
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

```

SL,FROM=S31USL2          =====>  SOURCE,SEARCH=SP2LIB. ...

```

```

* $$ JOB JNM=S31L503,CLASS=0,DISP=D
// JOB S31L503  DISPLAY E.WAIT
* SIPO 3.1 LIBRARIAN  ID=5  JOB=03
// LIBDEF SL,FROM=S31USL2
// EXEC ESERV
   DSPLY E.WAIT
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=SP2L503,CLASS=0,DISP=D
// JOB SP2L503  DISPLAY E.WAIT
* VSE/SP 2.1 LIBRARIAN  ID=5  JOB=03
// LIBDEF SOURCE,SEARCH=SP2LIB.USRSLB2
// EXEC ESERV
   DSPLY E.WAIT
/*
/&
* $$ EOJ

```

MAPPING OF PARAMETERS

```

SL,FROM=S31USL2          =====>  SOURCE,SEARCH=SP2LIB. ...

```

PHYSICAL LIBRARY MIGRATION

Old Libraries to New Libraries

```
* $$ JOB JNM=LIBMIG1,CLASS=0,DISP=D
// JOB LIBMIG1  BACKUP SIPO 3.1 PRIVATE LIBRARIES
* SIPO 3.1 LIBRARIAN
// ASSGN SYS006,1B0,D0
// MTC REW,SYS006
// EXEC BACKUP
  NOSA,NOSORT,NODSF
  BACKUP CL,LIB=(S31UCL1,S31UCL2)
  BACKUP RL,LIB=(S31URL1,S31URL2)
  BACKUP SL,LIB=(S31USL1,S31USL2)
  BACKUP PL,LIB=(S31UPL1,S31UPL2)
/*
// MTC WTM,SYS006,5
// MTC REW,SYS006
/*
/&
* $$ EOJ
```

```

* $$ JOB JNM=LIBMIG2,CLASS=0,DISP=D
// JOB LIBMIG2   DEFINE VSE/SP 2.1 TARGET LIBRARY INCL. SUBLIBS
* VSE/SP 2.1 LIBRARIAN
// DLBL SP2LIB,'USER.LIBRARY.SP2LIB',99/365,SD
// EXTENT ,SYSWK1,1,0,509938,48000
// EXEC LIBR
ON $RC > 0 GOTO LIBRERR
DEFINE L=SP2LIB R=Y
DEFINE S=SP2LIB.USRSLB1 -
      SP2LIB.USRSLB2 -
      REUSE=AUTO -
      R = Y
LD L=SP2LIB O=STAT
/. LIBRERR
/*
IF $MRC = 0 THEN
GOTO SUCCESS
* *****
* CHECK THE ERROR MESSAGE(S)
* IN CASE OF SYNTAX ERRORS - CORRECT STATEMENT(S) AND RERUN JOB
* *****
/. SUCCESS
// EXEC LISTLOG
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=LIBMIG3,CLASS=0,DISP=D
// JOB LIBMIG3  RESTORE SIPO 3.1 TO VSE/SP 2.1 TEMPORARY LIBRARY
* VSE/SP 2.1 LIBRARIAN
// DLBL SP2LIB,'USER.LIBRARY.SP2LIB',99/365,SD
// EXTENT ,SYSWK1,1,0,509938,48000
// ASSGN SYS006,1B3,D0
// MTC REW,SYS006
// EXEC LIBR
ON $RC > 0 GOTO LIBRERR
RESTORE OLD=S31UCL1:SP2LIB.USRSLB1 -
          S31URL1:SP2LIB.USRSLB1 -
          S31USL1:SP2LIB.USRSLB1 -
          S31UPL1:SP2LIB.USRSLB1 -
          S31UCL2:SP2LIB.USRSLB2 -
          S31URL2:SP2LIB.USRSLB2 -
          S31USL2:SP2LIB.USRSLB2 -
          S31UPL2:SP2LIB.USRSLB2 -
          T=SYS006 LIST=Y R=Y
/. LIBRERR
LD L=SP2LIB O=STAT
/*
IF $MRC = 0 THEN
GOTO SUCCESS
* *****
* CHECK THE ERROR MESSAGE(S)
* IN CASE OF SYNTAX ERRORS CORRECT STATEMENT AND RERUN JOB
*
* IN CASE OF LIBRARY FULL CONDITION, RERUN LIBRARY DEFINE JOBS
* (JNM=LIBMIG2) WITH LARGER EXTENT(S) AND THIS JOB
* *****
/. SUCCESS
/*
// MTC REW,SYS006
// EXEC LISTLOG
/*
/&
* $$ EOJ

```

Old Members to New Libraries

```
* $$ JOB JNM=OLDFORM,CLASS=0,DISP=D
// JOB OLDFORM PUNCH MACROS IN OLD FORMAT ONTO TAPE
// LIBDEF SL,FROM=S31USL2,
// ASSGN SYSPCH,1B3
// PAUSE DO YOU WANT TO OVERRIDE THE ASSIGNMENT?
// MTC REW,SYSPCH
// EXEC SSERV
// PUNCH A.RJEVERF,B.APP
/*
// MTC WTM,SYSPCH,3
// CLOSE SYSPCH
/&
* $$ EOJ
```

```
* $$ JOB JNM=NEWFORM,CLASS=6,DISP=D,NTFY=(,MUNDT),PRI=9,SYSID=1
* $$ LST CLASS=R,DEST=(,MUNDT),DISP=D
// JOB NEWFORM CATALOG OLD FORMAT MEMBERS INTO 2.1 LIBS
// ASSGN SYSIPT,1B3
// PAUSE DO YOU WANT TO OVERRIDE THE ASSIGNMENT?
// EXEC LIBR,PARM='ACC S=ISCLIB.HANS'
/*
// MTC REW,SYSIPT
/&
* $$ EOJ
```

```

* $$ JOB JNM=OLDPHASE,CLASS=0,DISP=D
// JOB OLDPHASE PUNCH PHASE IN OLD FORMAT ONTO TAPE
// LIBDEF CL,FROM=S31LUCL2,
// ASSGN SYSPCH,1B3
// PAUSE DO YOU WANT TO OVERRIDE THE ASSIGNMENT?
// MTC REW,SYSPCH
// EXEC CSERV
// PUNCH DFHPLP,DFHRTE
/*
// MTC WTM,SYSPCH,3
// MTC REW,SYSPCH
/&
* $$ EOJ

```

```

* $$ JOB JNM=NEWPHASE,CLASS=A,DISP=D,NTFY=(,MUNDT)
, PRI=9, SYSID=1
* $$ LST CLASS=R,DEST=(,MUNDT),DISP=D
// JOB NEWPHASE CATALOG OLD FORMAT PHASES INTO 2.1 LIBS
// ASSGN SYSIPT,1B3
// PAUSE DO YOU WANT TO OVERRIDE THE ASSIGNMENT?
// OPTION CATAL
// LIBDEF PHASE,CATALOG=ISCLIB.HANS
// INCLUDE
// EXEC LNKEDT
/*
// MTC REW,SYSIPT
/&
* $$ EOJ

```

New Libraries to Old Libraries

```
* $$ JOB JNM=PCHCL,CLASS=0,DISP=D
// JOB PCHCL  PHASES FOR BACKWARD MIGRATION
// ASSGN SYSPCH,1BO,DO
// MTC REW,SYSPCH
// EXEC LIBR
ACCESS SUBLIB=DELTA.SLIB
PUNCH *.PHASE FORMAT=OLD
/*
// MTC WTM,SYSPCH
// MTC RUN,SYSPCH
/*
/&
* $$ EOJ
```

```
* $$ JOB JNM=PCHRSP,CLASS=0,DISP=D
// JOB PCHRSP  OTHER MEMBER TYPES FOR BACKWARD MIGRATION
// ASSGN SYSPCH,1BO,DO
// MTC REW,SYSPCH
// EXEC LIBR
ACCESS SUBLIB=DELTA.SLIB
PUNCH *.OBJ FORMAT=OLD
PUNCH *.PROC FORMAT=OLD
PUNCH *.A,*.E FORMAT=OLD
/*
// MTC WTM,SYSPCH
// MTC RUN,SYSPCH
/*
/&
* $$ EOJ
```



```
* $$ JOB JNM=RECATPH,CLASS=0,DISP=D
// JOB RECATPH PHASES INTO "OLD LIBRARY"
// LIBDEF CL,TO=OLDCL
// OPTION CATAL
// ASSGN SYSIPT,1B0
// INCLUDE
/*
// EXEC LNKEDT
/&
* $$ EOJ
```

```
* $$ JOB JNM=RECATRSP,CLASS=0,DISP=D
// JOB RECATRSP OTHER MEMBER TYPES INTO "OLD LIBRARIES"
// LIBDEF RL,TO=OLDRL
// LIBDEF PL,TO=OLDPL
// LIBDEF SL,TO=OLDSL
// ASSGN SYSIPT,1B0
// EXEC MAINT
/*
/&
* $$ EOJ
```

LIBRARIES IN VSAM MANAGED SPACE

Definition

```
* $$ JOB JNM=VLIBCREA,CLASS=0,DISP=D,NTFY=YES
// JOB VLIBCREA DEFINE CLUSTER FOR LIBRARY IN VSAM MANAGED SPACE
// EXEC IDCAMS,SIZE=AUTO
  DEFINE CLUSTER ( -
    NAME (VSE.USERLIB.LIBRARY) -
    BLOCKS (16384 8192) -
    SHAREOPTIONS (3) -
    RECORDFORMAT (NOCIFORMAT) -
    VOLUMES (DOSRES SYSWK1 ) -
    NOREUSE -
    NONINDEXED -
    TO (99366)) -
    DATA (NAME (VSE.USERLIB.LIBRARY.DATA) -
    CATALOG (VSAM.MASTER.CATALOG) -
  )
  IF LASTCC NE 0 THEN CANCEL JOB
/*
// OPTION STDLABEL=ADD
// DLBL USERLIB,'VSE.USERLIB.LIBRARY',,VSAM,
      CAT=IJSYSCT,DISP=(OLD,KEEP)
/*
// EXEC LIBR,PARM='MSHP'
      DEFINE LIB=USERLIB REPLACE=YES
      DEFINE SUB=USERLIB.SUBLIB1 REUSE=IMM
/*
/&
* $$ EOJ
```

C

Extension

```
* $$ JOB JNM=VLIBEXT,CLASS=0,DISP=D,NTFY=YES
// JOB  VLIBEXT  EXTEND LIBRARY SPACE
// EXEC IDCAMS,SIZE=AUTO
  DEFINE SPACE ( -
    BLOCKS (32768 ) -
    ORIGIN (44240 ) -
    VOLUME (SYSWK2)) -
    CATALOG (VSAM.MASTER.CATALOG )
  IF LASTCC NE 0 THEN CANCEL JOB
  ALTER VSE.USERLIB.LIBRARY.DATA -
    ADDVOLUMES (SYSWK2) -
    CATALOG (VSAM.MASTER.CATALOG )
/*
/&
* $$ EOJ
```

LIBRARIAN MIGRATION REFERENCE

To enable a 'desk' check for not automatically migrated functions this chapter is subdivided in three parts:

- 'Non Migrated Functions',
- 'Migrated Functions with Restrictions',
- 'Automatically Migrated Functions'.

Subchapter one and three consist of two chapters:

- 'Job Control Functions'
- 'Librarian Services'.

The first section describes the JCL statements needed for executing Librarian functions, and the second section describes the different Librarian programs and their commands.

Each statement or function is described in a box. Each box consists of three areas titled:

- 'Pre-VSE/Advanced Functions 2.1 Function',
- 'Behavior under VSE/AF 2.1',
- 'Conversion Proposal'.

Non-Migrated Functions

This subchapter contains the list of all functions which are not migrated in any way. The following functions cause a failure of the old jobstream, and therefore must be changed.

Job Control Functions

In the following the not migrated JCL functions are described.

ASSGN SYSCLB

Pre-VSE/Advanced Functions 2.1 Statement:	
ASSGN SYSCLB,...	
Behavior under VSE/Advanced Functions 2.1:	
Message '1A43D INVALID LOGICAL UNIT SPECIFICATION.' Job waits for operator decision or is canceled if running as batch job with option NOLOG active. Operator action: new LIBDEF statements can be entered if option LOG is active.	
Conversion Proposal:	
For LNKEDT:	[[/]] LIBDEF PHASE,CATALOG=lib.sublib [,TEMP ,PERM]
For FETCH/LOAD:	[[/]] LIBDEF PHASE,SEARCH=lib.sublib [,TEMP ,PERM] -
	ASSGN SYSCLB,UA can be converted to [[/]] LIBDROP PHASE,CATALOG [,TEMP ,PERM] or [[/]] LIBDROP PHASE,SEARCH [,TEMP ,PERM]
For Librarian Services: not used anymore	

ASSGN SYSRLB

Pre-VSE/Advanced Functions 2.1 Statement:
ASSGN SYSRLB,...
Behavior under VSE/Advanced Functions 2.1:
Message '1A43D INVALID LOGICAL UNIT SPECIFICATION.' Job waits for operator decision or is canceled if running as batch job with option NOLOG active. Operator action: new LIBDEF statements can be entered if option LOG is active.
Conversion Proposal:
For LNKEDT: [//] LIBDEF OBJ,SEARCH=lib.sublib - [,TEMP ,PERM] [//] ASSGN SYSRLB,UA can be converted to [//] LIBDROP OBJ,SEARCH [,TEMP ,PERM] For Librarian Services: not used anymore

ASSGN SYSSLB

Pre-VSE/Advanced Functions 2.1 Statement:	
ASSGN SYSSLB,...	
Behavior under VSE/Advanced Functions 2.1:	
Message '1A43D INVALID LOGICAL UNIT SPECIFICATION.' Job waits for operator decision or is canceled if running as batch job with option NOLOG active. Operator action: new LIBDEF statements can be entered if option LOG is active.	
Conversion Proposal:	
For Compilations: [//] LIBDEF SOURCE,SEARCH=lib.sublib	-
[,TEMP ,PERM]	
For FETCH/LOAD: [//] LIBDEF PHASE,SEARCH=lib.sublib	-
[,TEMP ,PERM]	
[//] ASSGN SYSSLB, UA can be converted to	
[//] LIBDROP SOURCE,SEARCH [,TEMP ,PERM]	
For Librarian Services: not used anymore	

Librarian Services

In the following the not migrated Librarian functions are described.

CORGZ ALLOC

Pre-VSE/Advanced Functions 2.1 Statement:

```
// ASSGN SYS002,...
// DLBL IJSYSRS,...
// EXTENT SYS002,...
// EXEC CORGZ
  ALLOC CL=...,RL=...,SL=...,PL=....
  (followed by any COPY statement)
/*
```

Behavior under VSE/Advanced Functions 2.1:

JCL statements accepted.
CORGZ fails with message
 'L147I THE MIGRATION OF THE ALLOC STATEMENT IS
 NOT SUPPORTED'
followed by message
 '1S55I LAST RETURN CODE WAS 0016'.
The job is flushed and terminated with return code 16.

Conversion Proposal:

```
// DLBL IJSYSR1,...
// EXTENT
// EXEC LIBR
  DEFINE LIB=IJSYSR1
  [DEFINE SUBLIB=IJSYSR1.SYSLIB]
  (followed by any librarian command)
/*
```

Remarks: The library names IJSYSRn (n=1,2,...,9) specify non IPL'ed system libraries. IJSYSRS is the name of the currently IPL'ed system library.
The IPL bootstrap record is taken from the currently IPL'ed system library and stored to the new SYSRES automatically.

CORGZ NEWVOL

Pre-VSE/Advanced Functions 2.1 Statement:

```
// ASSGN SYSSLB,...  
// DLBL IJSYSSL,...  
// EXTENT SYSSLB,...  
// ASSGN SYSRLB,...  
// DLBL IJSYSRL,...  
// EXTENT SYSRLB,...  
// ASSGN SYS003,...  
// DLBL IJSYSPC,...  
// EXTENT SYS003,...  
// EXEC CORGZ  
  NEWVOL CL=...,RL=...,SL=...  
  (followed by any COPY statement)  
/*
```

Behavior under VSE/Advanced Functions 2.1:

ASSGN SYSSLB and ASSGN SYSRLB are rejected with message 1A43D (see also chapter 'Job Control Functions'). Job waits for operator decision.

Operator action: Cancel the job.

If the operator doesn't cancel and types EOB then:

CORGZ fails with message

'L147I THE MIGRATION OF THE NEWVOL STATEMENT IS
NOT SUPPORTED'

followed by message

'1S55I LAST RETURN CODE WAS 0016'.

The job is flushed and terminated with return code 16.

Conversion Proposal:

```
// EXEC LIBR
[DEFINE LIB=lib]
  DEFINE SUBLIB=lib.sublib1[,...]
  (followed by any librarian command)
/*
```

Remark:

The new Library conception allows storing of different member types into the same sublibrary. That means it is no longer necessary to create a new type dependent sublibrary by analogy with the old libraries Core Image, Procedure, Relocatable, and Source Statement. Therefore it should be considered if the CORGZ NEWVOL step can be erased due to a better organization of the members in fewer sublibraries.

CORGZ COPYI

Pre-VSE/Advanced Functions 2.1 Statement:
<pre>(JCL statements: LIBDEF, ASSGN SYSCLB) // EXEC CORGZ COPYI ... /*</pre>
Behavior under VSE/Advanced Functions 2.1:
(JCL statements LIBDEF, ASSGN SYSCLB - see 'Job Control Functions') CORGZ fails with message 'L147I THE MIGRATION OF THE COPYI STATEMENT IS NOT SUPPORTED' followed by message '1S55I LAST RETURN CODE WAS 0016'. The job is flushed and terminated with return code 16.
Conversion Proposal:
Not used anymore

CORGZ COPY ALL

Pre-VSE/Advanced Functions 2.1 Statement:

```
( JCL statements: LIBDEF, ASSGN SYSCLB|SYSRLB|SYSSLB )  
  
// EXEC CORGZ  
   COPY ALL  
/*
```

Behavior under VSE/Advanced Functions 2.1:

(JCL statements LIBDEF, ASSGN SYSCLB - see
'Job Control Functions')

CORGZ fails with message
'L147I THE MIGRATION OF THE COPY STATEMENT IS NOT SUPPORTED'
followed by message
'1S55I LAST RETURN CODE WAS 0016'.
The job is flushed and terminated with return code 16.

Conversion Proposal:

```
// EXEC LIBR  
   CONNECT SUBLIB=lib1.sublib1:lib2.sublib2  
   COPY *.*  
/*
```

DSERV CD (PHASENAME ..)

Pre-VSE/Advanced Functions 2.1 Statement:

(JCL statements: LIBDEF, ASSGN SYSCLB)

```
// EXEC DSERV
  DSPLY[S] CD(phaseName[,nn])
/*
```

Behavior under VSE/Advanced Functions 2.1:

(JCL statements LIBDEF,ASSGN SYSCLB - see
'Job Control Functions')

DISPLY[S] CD(phaseName[,nn]) fails with message
'L011I INVALID SYNTAX - - - (phaseName[,nn]) and
processing continues with next DSERV DSPLY command if available.

Conversion Proposal:

```
// EXEC LIBR
  ACCESS SUBLIB=lib.sublib
  LISTDIR phaseName.PHASE
/*
```

Remark:

The display of the version/modification level or of the level
identification is no longer supported.

MAINT ALLOC

Pre-VSE/Advanced Functions 2.1 Statement:
The ALLOC command was already dropped for VSE/Advanced Functions Release 2.
Behavior under VSE/Advanced Functions 2.1:
The ALLOC command fails with message 'L010I INVALID COMMAND NAME' Processing goes on with next MAINT command if available. At end of step JCL issues the message '1S55I LAST RETURN CODE WAS 0008'.
Conversion Proposal:
Not used anymore

MAINT UPDATE

Pre-VSE/Advanced Functions 2.1 Statement:

```
(JCL statements LIBDEF,ASSGN SYSSLB )  
  
// EXEC MAINT  
  UPDATE A.name1[,A.name2][,v.m][,nn]  
    ( subcommands + data records )  
/*
```

Behavior under VSE/Advanced Functions 2.1:

(JCL statements LIBDEF, ASSGN SYSSLB see chapter
'Job Control Functions'.)

Operator action for failing JCL statements:

Cancel job because the following MAINT/UPDATE fails.

The UPDATE command is not automatically migrated.
The UPDATE command and all subsequent statements up to the ')END'
statement fail with message
'L010I INVALID COMMAND NAME'
Processing goes on with next MAINT command if available. At end of
step JCL issues the message
'1S55I LAST RETURN CODE WAS 0008'.

MAINT UPDATE (cont'd)

Conversion Proposal:

```
// EXEC LIBR
ACCESS SUBLIB=lib.sublib
UPDATE name1.A [SAVE=name2.A] [SEQUENCE=n|FS|NO]      -
               [COLUMN=start:end]
               /* old UPDATE A.name1 ...            */
( Subcommands are same as for the old UPDATE.
  Exception: the operands of the ') END' statement are no
  longer supported. )
/*
```

BACKUP

Pre-VSE/Advanced Functions 2.1 Statement:

(JCL statements)

```
// EXEC BACKUP
      (control statements)
/*
```

Behavior under VSE/Advanced Functions 2.1:

The old program BACKUP is not migrated automatically.
An information is printed on SYSLSST, the job is flushed and
terminated with return code 16.

Conversion Proposal:

1. Backup of complete libraries:

```
// EXEC LIBR
      BACKUP LIB=lib1 [lib2 ...] TAPE=SYSnnn|cuu -
              [RESTORE=ONLINE|STANDALONE]
/*
```

2. Backup of sublibraries:

```
// EXEC LIBR
      BACKUP SUBLIB=lib1.sublib1 [lib2.sublib2 ...] TAPE=SYSnnn|cuu
/*
```

RESTORE

Pre-VSE/Advanced Functions 2.1 Statement:

(JCL statements)

```
// EXEC RESTORE
      (control statements)
/*
```

Behavior under VSE/Advanced Functions 2.1:

The old program RESTORE is not migrated automatically.
An information is printed on SYSLSST, the job is flushed and
terminated with return code 16.

RESTORE (CONTD.)

Conversion Proposal:

1. Restore of complete libraries:

```
// EXEC LIBR
  RESTORE LIB=lib1[:newlib1] [lib2[: newlib2] ..] TAPE=SYSnnn|cuu-
    [REPLACE=YES|NO]
/*
```

2. Restore of sublibraries:

```
// EXEC LIBR
  RESTORE SUBLIB=lib1.sublib1[:newlib1.newslib1]           -
    [lib2.sublib2[:newlib2.sublib2] ...]                   -
    TAPE=SYSnnn|cuu [REPLACE=YES|NO]
/*
```

3. Restore of pre-VSE/Advanced Functions 2.1 tapes:

```
// EXEC LIBR
  RESTORE OLDLIB=oldlib1:newlib1.newslib1                 -
    [oldlib2:newlib2.newslib2 ...]                         -
    TAPE=SYSnnn|cuu [REPLACE=YES|NO]
/*
```

or

4. Restore of single members into different sublibraries:

```
// EXEC LIBR
  ACCESS lib.sublib /* used if no target lib.sublib -
    specification in the RESTORE command */
  RESTORE lib.sublib.name1.type1[:lib1.sublib1]           -
    lib.sublib.name2.type2[:lib2.sublib2]...              -
    TAPE=SYSnnn|cuu [REPLACE=YES|NO]
/*
```

Remark:

See the specification of the RESTORE command for further Restore functions.

COPYSERV

Pre-VSE/Advanced Functions 2.1 Statement:

```
( JCL statements - ASSGN SYSCLB|SYSRLB|SYSSLB)
// EXEC COPYSERV
/ &
```

Behavior under VSE/Advanced Functions 2.1:

```
( JCL statements ASSGN SYSCLB|SYSRLB|SYSSLB see chapter
'Job Control Functions')
// EXEC COPYSERV fails with message '1U53A PROGRAM NOT FOUND'.
Job waits for operator action or is canceled if
running as batch job with option NOLOG active.
```

Operator action: Cancel the job.

Conversion Proposal:

```
// EXEC LIBR

/* RES,NRS :                               */
COMPARE SUBLIB=IJSYSRS.SYSLIB:IJSYSRn.SYSLIB PUNCH=YES

/* RES,PRV : UPSI 00001000 (SYSCLB) */
CONNECT SUBLIB=IJSYSRS.SYSLIB:lib.sublib
COMPARE *.PHASE PUNCH=YES

/* RES,PRV : UPSI 00000100 (SYSRLB) */
CONNECT SUBLIB=IJSYSRS.SYSLIB:lib.sublib
COMPARE *.OBJ PUNCH=YES

/* RES,PRV : UPSI 00000010 (SYSSLB) */
CONNECT SUBLIB=IJSYSRS.SYSLIB:lib.sublib
COMPARE *.A [*.E ... ] PUNCH=YES /* specify all source types -
                                   contained in IJSYSRS.SYSLIB */
```

COPYSERV (CONTD.)

```
/* PRV,NRS : UPSI 10000000 (SYSCLB) */
CONNECT SUBLIB=lib.sublib:IJSYSRn.SYSLIB
COMPARE *.PHASE PUNCH=YES

/* PRV,NRS : UPSI 01000000 (SYSRLB) */
CONNECT SUBLIB=lib.sublib:IJSYSRn.SYSLIB
COMPARE *.OBJ PUNCH=YES

/* PRV,NRS : UPSI 00100000 (SYSSLB) */
CONNECT SUBLIB=lib.sublib:IJSYSRn.SYSLIB
COMPARE *.A [*.E ... ] PUNCH=YES /* specify all source types -
                                   contained in lib.sublib */

/* PRV,PRV : UPSI 10001000 (SYSCLB) */
CONNECT SUBLIB=lib1.sublib1:lib2.sublib2
COMPARE *.PHASE PUNCH=YES

/* PRV,PRV : UPSI 01000100 (SYSRLB) */
CONNECT SUBLIB=lib1.sublib1:lib2.sublib2
COMPARE *.OBJ PUNCH=YES

/* PRV,PRV : UPSI 00100010 (SYSSLB) */
CONNECT SUBLIB=lib1.sublib1:lib2.sublib2
COMPARE *.A [*.E ... ] PUNCH=YES /* specify all source types -
                                   contained in lib1.sublib1 */
/*

(n = 1, 2, ... ,9 used for name IJSYSRn )
```

COPYSERV (CONTD.)

Remark:

If the different member types are stored in one sublibrary
the COPYSERV conversion for private libraries can be done as follows:

```
// EXEC LIBR
  COMPARE SUBLIB=IJSYSRS.SYSLIB:lib.sublib PUNCH=YES /* RES,PRV */
  COMPARE SUBLIB=lib.sublib:IJSYSRn.SYSLIB PUNCH=YES /* PRV,NRS */
  COMPARE SUBLIB=lib1.sublib1:lib2.sublib2 PUNCH=YES /* PRV,PRV */
/*
```

Consider that there exists also a COPY function with the REPLACE=NO
parameter as default, that is only those members are copied which
are not yet contained in the 'to-sublibrary':

```
// EXEC LIBR
  COPY SUBLIB=IJSYSRS.SYSLIB:lib.sublib /* RES,PRV */
  COPY SUBLIB=lib.sublib:IJSYSRn.SYSLIB /* PRV,NRS */
  COPY SUBLIB=lib1.sublib1:lib2.sublib2 /* PRV,PRV */
/*
```


PDZAP

Pre-VSE/Advanced Functions 2.1 Statement:
<pre>[//] EXEC PDZAP</pre>
Behavior under VSE/Advanced Functions 2.1:
Message '1U53A PROGRAM NOT FOUND'. Job waits for operator action or is canceled if running as batch job with option NOLOG active. Operator action: Cancel the job.
Conversion Proposal:
Example: <pre>// EXEC MSHP CORRECT 5745-SC-SUP:LF00001 AFFECTS PHASE=\$\$a\$sup1 ALTER 0040 4700:47F0 /*</pre>

Migrated Functions with Restrictions

This chapter lists functions which are migrated. But there are either functional restrictions (// OPTION LINK) which have to be considered, or LIBDEF statements are required/must be changed in order to run the job successfully.

ESERV

Pre-VSE/Advanced Functions 2.1 Statement:

```
// LIBDEF SL,FROM=... or // ASSGN SYSSLB    see chapter  
                                           'Job Control Functions'  
// EXEC ESERV  
  (any command)  
/*
```

Behavior under VSE/Advanced Functions 2.1:

ESERV works with the SEARCH chain only, that is a LIBDEF FROM is no longer accessed by ESERV, even if the library has an entry in the Migration Table. If no LIBDEF SEARCH is available ESERV accesses the system sublibrary IJSYSRS.SYSLIB as default.

Conversion Proposal:

```
// LIBDEF SOURCE,SEARCH= ...  
// EXEC ESERV  
  (any command)  
/*
```

Remark:

The externals of the ESERV program do not change.

LNKEDT

Pre-VSE/Advanced Functions 2.1 Statement:	
// EXEC LNKEDT	
Behavior under VSE/Advanced Functions 2.1:	
MIG.TABLE YES	LIBDEF RL,SEARCH ... and LIBDEF CL,TO ... are processed by LNKEDT automatically. They can be mixed with new LIBDEF OBJ and LIBDEF PHASE statements. LNKEDT processes always the active LIBDEF statement either old or new. Restriction: LIBDEF CL,TO=... or LIBDEF PHASE,CATALOG=... is required for OPTION CATAL otherwise the LNKEDT step will be cancelled with message: 2195I NO TARGET CATALOG SUBLIBRARY SPECIFIED
MIG.TABLE NO:	LNKEDT requires active LIBDEF PHASE,CATALOG ... for OPTION CATAL and optionally LIBDEF OBJ,SEARCH ...
Conversion Proposal:	
The LNKEDT program externals are unchanged. Change old LIBDEF RL,SEARCH=... to new LIBDEF OBJ,SEARCH=... and old LIBDEF CL,TO=... to new LIBDEF PHASE,CATALOG=... Change ASSGN SYSCLB SYSRLB to LIBDEF PHASE,CATALOG=... and LIBDEF OBJ,SEARCH=... (see also restriction for // OPTION LINK).	

OPTION LINK

Pre-VSE/Advanced Functions 2.1 Statement:
// OPTION LINK
Behavior under VSE/Advanced Functions 2.1:
accepted Restriction: LNKEDT step can process 1 phase only when running in temporary linkage mode. Otherwise the step is canceled.
Conversion Proposal:
Use // OPTION CATAL when linking more than 1 phase in the same LNKEDT step and delete the phases afterwards. When in test mode use a catalog-sublibrary with attribute IMMEDIATE.

Automatically Migrated Functions

This chapter lists all functions which are migrated automatically. The main idea of this chapter is to have conversion proposals for migrated functions, too. Furthermore it can be used as a quick reference about the behavior of migrated functions.

Job Control Functions

Note: 'MIG. TABLE YES' means: There exists an entry in the Migration Table INLPLMT for the old library name.

'MIG. TABLE NO' means: There exists no entry in the Migration Table INLPLMT for the old library name or the Migration Table does not exist.

LIBDEF FROM

Pre-VSE/Advanced Functions 2.1 Statement:	
[//] LIBDEF {CL RL SL PL},FROM=lib1 [,TEMP ,PERM]	
Behavior under VSE/Advanced Functions 2.1:	
MIG.TABLE YES:	accepted
MIG.TABLE NO:	Message '1D13D CONVERSION FOR lib1 FAILED' Job waits for operator decision. The statement can be ignored if followed by CORGZ/CSERV/DSERV/PSERV/RSERV/SSERV - these programs prompt the operator if LIBDEF FROM is missing. To avoid prompting an entry in the Migration Table has to be specified. (There is no default for the system library IJSYSRS.)
Conversion Proposal:	
Not used anymore for VSE/Advanced Functions 2.1 functions	

LIBDEF NEW

Pre-VSE/Advanced Functions 2.1 Statement:
<code>[//] LIBDEF {CL RL SL PL},NEW=lib1 [,TEMP]</code>
Behavior under VSE/Advanced Functions 2.1:
Ignored without message. The statement can be deleted since there is no migration support for CORGZ NEWVOL.
Conversion Proposal:
Not used anymore for VSE/Advanced Functions 2.1 functions

LIBDEF TO

Pre-VSE/Advanced Functions 2.1 Statement:	
[//] LIBDEF {CL RL SL PL},TO=lib1 [,TEMP ,PERM]	
Behavior under VSE/Advanced Functions 2.1:	
MIG.TABLE YES:	accepted
MIG.TABLE NO:	Message '1D13D CONVERSION FOR lib1 FAILED' Job waits for operator decision or is canceled if running as batch job with option NOLOG active. The statement can be ignored if followed by CORGZ/MAINT. These programs prompt the operator if LIBDEF TO is missing. To avoid prompting an entry in the Migration Table has to be specified. For LNKEDT the Migration Table is required. (There is no default for the system library IJSYSRS.)
Conversion Proposal:	
LNKEDT only: [//] LIBDEF PHASE,CATALOG=lib.sublib1 [,TEMP ,PERM] other functions: not applicable for VSE/Advanced Functions 2.1	

LIBDEF SEARCH

Pre-VSE/Advanced Functions 2.1 Statement:	
[//] LIBDEF {CL RL SL PL},SEARCH=(lib1,lib2,...) [,TEMP ,PERM]	
Behavior under VSE/Advanced Functions 2.1:	
MIG.TABLE YES:	accepted
MIG.TABLE NO:	Message '1D13D CONVERSION FOR lib1 FAILED' Job waits for operator decision or is canceled if running as batch job with option NOLOG active.The Migration Table is required. (This is also valid for library name IJSYSRS.)
Conversion Proposal:	
[//] LIBDEF {PHASE OBJ SOURCE PROC}, SEARCH=(lib.sublib1,lib.sublib2,...) [,TEMP ,PERM]	
C	

LIBDROP FROM

Pre-VSE/Advanced Functions 2.1 Statement:
<code>[//] LIBDROP {CL RL SL PL},FROM [,TEMP ,PERM]</code>
Behavior under VSE/Advanced Functions 2.1:
accepted
Conversion Proposal:
Not used anymore for VSE/Advanced Functions 2.1 functions

LIBDROP TO

Pre-VSE/Advanced Functions 2.1 Statement:
[//] LIBDROP {CL RL SL PL},TO [,TEMP ,PERM]
Behavior under VSE/Advanced Functions 2.1:
accepted
Conversion Proposal:
LNKEDT only: [//] LIBDROP PHASE,CATALOG [,TEMP ,PERM] Librarian Services: not used anymore

LIBDROP NEW

Pre-VSE/Advanced Functions 2.1 Statement:
<code>[//] LIBDROP {CL RL SL PL},NEW [,TEMP]</code>
Behavior under VSE/Advanced Functions 2.1:
Ignored without message
Conversion Proposal:
not used anymore for VSE/Advanced Functions 2.1 Functions

LIBDROP SEARCH

Pre-VSE/Advanced Functions 2.1 Statement:
[//] LIBDROP {CL RL SL PL},SEARCH [,TEMP ,PERM}
Behavior under VSE/Advanced Functions 2.1:
accepted
Conversion Proposal:
[//] LIBDROP {PHASE OBJ SOURCE PROC},SEARCH [,TEMP ,PERM]

LIBDROP ALL

Pre-VSE/Advanced Functions 2.1 Statement:
[//] LIBDROP {CL RL SL PL},ALL [,TEMP ,PERM]
Behavior under VSE/Advanced Functions 2.1:
accepted
Conversion Proposal:
[//] LIBDROP {PHASE OBJ SOURCE PROC} [,TEMP ,PERM] Remark: There is a new function to free all LIBDEF's (that is all 4 types): [//] LIBDROP * [,TEMP ,PERM]

LIBLIST BG|FN

Pre-VSE/Advanced Functions 2.1 Statement:
[//] LIBLIST {CL RL SL PL} {,BG ,F} [,SYSLST ,SYSLOG]
Behavior under VSE/Advanced Functions 2.1:
accepted
Conversion Proposal:
[//] LIBLIST {PHASE OBJ SOURCE PROC} {,BG ,Fn} [,SYSLST ,SYSLOG]

LIBLIST [*]

Pre-VSE/Advanced Functions 2.1 Statement:
[//] LIBLIST {CL RL SL PL} [,*] [,SYSLST ,SYSLOG]
Behavior under VSE/Advanced Functions 2.1:
accepted
Conversion Proposal:
[//] LIBLIST {PHASE OBJ SOURCE PROC} [,SYSLST ,SYSLOG]

LIBLIST ALL

Pre-VSE/Advanced Functions 2.1 Statement:
[//] LIBLIST {CL RL SL PL},ALL [,SYSLST ,SYSLOG]
Behavior under VSE/Advanced Functions 2.1:
accepted
Conversion Proposal:
[//] LIBLIST {PHASE OBJ SOURCE PROC},* [,SYSLST ,SYSLOG] Remark: There is a new function to list all LIBDEF's (that is all 4 types): [//] LIBLIST * [,BG ,Fn ,*] [,SYSLST ,SYSLOG]

EXEC CORGZ

Pre-VSE/Advanced Functions 2.1 Statement:
<pre>[// UPSI 1] [//] EXEC CORGZ</pre>
Behavior under VSE/Advanced Functions 2.1:
accepted
Conversion Proposal:
<pre>[//] EXEC LIBR [CONNECT command] COPY command (see Note) /*</pre>

Note: For detailed information, see "Librarian Services" on page 244.

EXEC DSERV

Pre-VSE/Advanced Functions 2.1 Statement:
[//[EXEC DSERV
Behavior under VSE/Advanced Functions 2.1:
accepted
Conversion Proposal:
[//[EXEC LIBR LISTDIR command (see Note) /*

Note: For detailed information, see "DSERV" on page 280.

EXEC CSERV

Pre-VSE/Advanced Functions 2.1 Statement:
<pre>[//] EXEC CSERV</pre>
Behavior under VSE/Advanced Functions 2.1:
accepted
Conversion Proposal:
<pre>[//] EXEC LIBR ACCESS command LIST PUNCH command (see Note) /*</pre>

Note: For detailed information, see "CSERV DSPLY" on page 256.

EXEC PSERV

Pre-VSE/Advanced Functions 2.1 Statement:
[//] EXEC PSERV
Behavior under VSE/Advanced Functions 2.1:
accepted
Conversion Proposal:
[//] EXEC LIBR ACCESS command LIST PUNCH command (see Note) /*

Note: For detailed information, see "PSERV DSPLY" on page 262.

EXEC RSERV

Pre-VSE/Advanced Functions 2.1 Statement:
<code>[//] EXEC RSERV</code>
Behavior under VSE/Advanced Functions 2.1:
accepted
Conversion Proposal:
<code>[//] EXEC LIBR ACCESS command LIST PUNCH command (see Note) /*</code>

Note: For detailed information, see "RSERV DSPLY" on page 268.

EXEC SSERV

Pre-VSE/Advanced Functions 2.1 Statement:
[//] EXEC SSERV
Behavior under VSE/Advanced Functions 2.1:
accepted
Conversion Proposal:
[//] EXEC LIBR ACCESS command LIST PUNCH command (see Note) /*

Note: For detailed information, see "SSERV DSPLY" on page 274.

EXEC MAINT

Pre-VSE/Advanced Functions 2.1 Statement:
[//] EXEC MAINT
Behavior under VSE/Advanced Functions 2.1:
accepted
Conversion Proposal:
[//] EXEC LIBR ACCESS command CATALOG DELETE RENAME command (see Note) /*

Note: For detailed information, see "MAINT CATALP" on page 283.

Librarian Services

Note: 'LIBDEF YES' means: The necessary old LIBDEF statements were processed successfully and are therefore active.

'LIBDEF NO' means: The necessary old LIBDEF statements were not processed successfully or are missing and are therefore inactive.

The Librarian Services prompt for missing LIBDEF information. Once a lib.sublib is entered for any type CL|PL|RL|SL it remains active for the complete step.

CORGZ MERGE-COPY OVERVIEW

Pre-VSE/Advanced Functions 2.1 Statement:	
(JCL statements: LIBDEF, ASSGN SYSCLB SYSRLB SYSSLB) <pre>// EXEC CORGZ MERGE ... COPYC ... COPYP ... COPYR ... COPYS ... /*</pre>	
Behavior under VSE/Advanced Functions 2.1:	
(JCL statements LIBDEF, ASSGN SYSCLB SYSRLB SYSSLB see 'Job Control Functions'.) Operator action for failing JCL statements: Type EOB in order to continue processing. The CORGZ migration program prompts for the missing lib.sublib information.	
LIBDEF YES:	MERGE is ignored without message COPYC is accepted and migrated automatically COPYP is accepted and migrated automatically COPYR is accepted and migrated automatically COPYS is accepted and migrated automatically

CORGZ MERGE-COPY OVERVIEW (CONTD.)

LIBDEF NO:	MERGE is ignored without message COPYC prompts for missing from- and/or to-lib.sublib COPYP prompts for missing from- and/or to-lib.sublib COPYR prompts for missing from- and/or to-lib.sublib COPYS prompts for missing from- and/or to-lib.sublib The prompting messages for COPYC are: 'L023I LIBDEF MISSING FOR FROM - LIBRARY OF TYPE CL' 'L146A ENTER THE LIBRARY.SUBLIBRARY TO BE PROCESSED - OR TYPE CANCEL' 'L024I LIBDEF MISSING FOR TO - LIBRARY OF TYPE CL' 'L146A ENTER THE LIBRARY.SUBLIBRARY TO BE PROCESSED - OR TYPE CANCEL' The prompting messages for COPYP, COPYR, COPYS are the same with different types PL RL SL instead of CL.
Conversion Proposal:	
<pre>// EXEC LIBR CONNECT SUBLIB=lib1.sublib1:lib2.sublib2 COPY name1.PHASE [name2.PHASE ...] /* used for COPYC */ COPY name1.PROC [name2.PROC ...] /* used for COPYP */ COPY name1.OBJ [name2.OBJ ...] /* used for COPYR */ COPY name1.A [name2.E ...] /* used for COPYS */ /*</pre> The UPSI flag for member lists has to be converted to the LIST=YES parameter on the new COPY statement. Remark: The new Library conception allows storing of different member types into the same sublibrary. That means it is no longer necessary to have different type dependent sublibraries by analogy with the old libraries Core Image, Procedure, Relocatable, and Source Statement.	

CORGZ MERGE

Pre-VSE/Advanced Functions 2.1 Statement:

```
(JCL statements: LIBDEF, ASSGN SYSCLB|SYSRLB|SYSSLB )  
  
// EXEC CORGZ  
  MERGE ...  
    (followed by COPY statement)  
/*
```

Behavior under VSE/Advanced Functions 2.1:

(JCL statements LIBDEF, ASSGN SYSCLB|SYSRLB|SYSSLB see
'Job Control Functions'.)

Operator action for failing JCL statements:

Type EOB in order to continue processing
if the CORGZ step contains MERGE, COPYC, COPYP, COPYR, COPYS
statements only. The CORGZ migration program prompts for the
missing lib.sublib information.

Cancel the job

if the CORGZ step contains COPYI, COPY ALL statements which
cause the failure of the step.

Each MERGE statement is ignored without message.

Conversion Proposal:

```
// EXEC LIBR  
  CONNECT SUBLIB=lib1.sublib1:lib2.sublib2  
    (followed by any COPY member statement)  
/*
```

Remark:

RES is now IJSYSRS.SYSLIB

NRS is now IJSYSRn.SYSLIB (n = 1, 2, ... ,9)

PRV is now any lib.sublib (except IJSYSRS.SYSLIB, IJSYSRn.SYSLIB)

CORGZ COPYC

Pre-VSE/Advanced Functions 2.1 Statement:	
<pre>(JCL statements: LIBDEF, ASSGN SYSCLB) // EXEC CORGZ MERGE ... COPYC name1[,name2,...] COPYC prog.ALL[,prog.ALL,...] COPYC ALL COPYC NEW /*</pre>	
Behavior under VSE/Advanced Functions 2.1:	
(JCL statements LIBDEF, ASSGN SYSCLB see 'Job Control Functions'.) Operator action for failing JCL statements: Type EOB in order to continue processing. The CORGZ migration program prompts for the missing lib.sublib information.	
LIBDEF YES:	<pre>MERGE ... : ignored without message COPYC name1[,name2,...] : accepted COPYC prog.ALL[,prog.ALL,...] : accepted COPYC ALL : accepted COPYC NEW : accepted</pre>
LIBDEF NO:	<pre>MERGE ... : ignored without message The first COPYC command causes the following prompting: 'L023I LIBDEF MISSING FOR FROM - LIBRARY OF TYPE CL' 'L146A ENTER THE LIBRARY.SUBLIBRARY TO BE PROCESSED - OR TYPE CANCEL' 'L024I LIBDEF MISSING FOR TO - LIBRARY OF TYPE CL' 'L146A ENTER THE LIBRARY.SUBLIBRARY TO BE PROCESSED - OR TYPE CANCEL' After valid lib.sublib specifications are entered all COPYC statements are accepted and migrated like above under 'LIBDEF YES'.</pre>

CORGZ COPYC (CONTD.)

Conversion Proposal:

```
// EXEC LIBR
  CONNECT SUBLIB=lib1.sublib1:lib2.sublib2
  COPY name1.PHASE [name2.PHASE ...] /* old COPYC name1 ... */
  COPY prog*.PHASE [prog*.PHASE ...] /* old COPYC prog.ALL ...*/
  COPY *.PHASE REPLACE=YES           /* old COPYC ALL          */
  COPY *.PHASE REPLACE=NO            /* old COPYC NEW         */
```

/*

Remarks:

REPLACE=NO is default - this fact has to be considered when converting to new COPY statements. The automatic migration uses REPLACE=YES. The list of copied members can be triggered by LIST=YES. The UPSI statement is no longer used. Generic specification is generalized, it is no longer restricted to the first 4 characters of a phase name.

CORGZ COPYP

Pre-VSE/Advanced Functions 2.1 Statement:	
<pre> (JCL statements: LIBDEF) // EXEC CORGZ MERGE ... COPYP name1[,name2,...] COPYP ALL COPYP NEW /* </pre>	
Behavior under VSE/Advanced Functions 2.1:	
(JCL statements LIBDEF see 'Job Control Functions'.) Operator action for failing JCL statements: Type EOB in order to continue processing. The CORGZ migration program prompts for the missing lib.sublib information.	
LIBDEF YES:	<pre> MERGE ... : ignored without message COPYP name1[,name2,...] : accepted COPYP ALL : accepted COPYP NEW : accepted </pre>
LIBDEF NO:	<pre> MERGE ... : ignored without message The first COPYP command causes the following prompting: 'L023I LIBDEF MISSING FOR FROM - LIBRARY OF TYPE PL' 'L146A ENTER THE LIBRARY.SUBLIBRARY TO BE PROCESSED - OR TYPE CANCEL' 'L024I LIBDEF MISSING FOR TO - LIBRARY OF TYPE PL' 'L146A ENTER THE LIBRARY.SUBLIBRARY TO BE PROCESSED - OR TYPE CANCEL' After valid lib.sublib specifications are entered all COPYP statements are accepted and migrated like above under 'LIBDEF YES'. </pre>

CORGZ COPYP (CONTD.)

Conversion Proposal:

```
// EXEC LIBR
  CONNECT SUBLIB=lib1.sublib1:lib2.sublib2
  COPY name1.PROC [name2.PROC ...]      /* old COPYP name1 ... */
  COPY *.PROC REPLACE=YES               /* old COPYP ALL      */
  COPY *.PROC REPLACE=NO                /* old COPYP NEW      */
/*
```

Remarks:

REPLACE=NO is default - this fact has to be considered when converting to new COPY statements. The automatic migration uses REPLACE=YES.

The list of members copied can be triggered by LIST=YES. The UPSI statement is no longer used.

CORGZ COPYR

Pre-VSE/Advanced Functions 2.1 Statement:	
<pre> (JCL statements: LIBDEF, ASSGN SYSRLB) // EXEC CORGZ MERGE ... COPYR name1[,name2,...] COPYR prog.ALL[,prog.ALL,...] COPYR ALL COPYR NEW /* </pre>	
Behavior under VSE/Advanced Functions 2.1:	
(JCL statements LIBDEF, ASSGN SYSRLB see 'Job Control Functions'.) Operator action for failing JCL statements: Type EOB in order to continue processing. The CORGZ migration program prompts for the missing lib.sublib information.	
LIBDEF YES:	<pre> MERGE ... : ignored without message COPYR name1[,name2,...] : accepted COPYR prog.ALL[,prog.ALL,...] : accepted COPYR ALL : accepted COPYR NEW : accepted </pre>
LIBDEF NO:	<pre> MERGE ... : ignored without message The first COPYR command causes the following prompting: 'L023I LIBDEF MISSING FOR FROM - LIBRARY OF TYPE RL' 'L146A ENTER THE LIBRARY.SUBLIBRARY TO BE PROCESSED - OR TYPE CANCEL' 'L024I LIBDEF MISSING FOR TO - LIBRARY OF TYPE RL' 'L146A ENTER THE LIBRARY.SUBLIBRARY TO BE PROCESSED - OR TYPE CANCEL' After valid lib.sublib specifications are entered all COPYR statements are accepted and migrated like above under 'LIBDEF YES'. </pre>

CORGZ COPYR (CONTD.)

Conversion Proposal:

```
// EXEC LIBR
  CONNECT SUBLIB=lib1.sublib1:lib2.sublib2
  COPY name1.OBJ [name2.OBJ ...]      /* old COPYR name1 ... */
  COPY prog*.OBJ [prog*.OBJ ...]     /* old COPYR prog.ALL ...*/
  COPY *.OBJ REPLACE=YES             /* old COPYR ALL      */
  COPY *.OBJ REPLACE=NO              /* old COPYR NEW      */
/*
```

Remarks:

REPLACE=NO is default - this fact has to be considered when converting to new COPY statements. The automatic migration uses REPLACE=YES. The list of copied members can be triggered by LIST=YES. The UPSI statement is no longer used. Generic specification is generalized, it is no longer restricted to the first 3 characters of an object deck name.

CORGZ COPYS

Pre-VSE/Advanced Functions 2.1 Statement:	
<pre> (JCL statements: LIBDEF, ASSGN SYSSLB) // EXEC CORGZ MERGE ... COPYS A.name1[,E.name2,...] COPYS A.ALL[,E.ALL,...] COPYS ALL COPYS NEW /* </pre>	
Behavior under VSE/Advanced Functions 2.1:	
(JCL statements LIBDEF, ASSGN SYSSLB see 'Job Control Functions'.) Operator action for failing JCL statements: Type EOB in order to continue processing. The CORGZ migration program prompts for the missing lib.sublib information.	
LIBDEF YES:	<pre> MERGE ... : ignored without message COPYS A.name1[,E.name2,...] : accepted COPYS A.ALL[,E.ALL,...] : accepted COPYS ALL : accepted COPYS NEW : accepted </pre>
LIBDEF NO:	<pre> MERGE ... : ignored without message The first COPYS command causes the following prompting: 'L023I LIBDEF MISSING FOR FROM - LIBRARY OF TYPE SL' 'L146A ENTER THE LIBRARY.SUBLIBRARY TO BE PROCESSED - OR TYPE CANCEL' 'L024I LIBDEF MISSING FOR TO - LIBRARY OF TYPE SL' 'L146A ENTER THE LIBRARY.SUBLIBRARY TO BE PROCESSED - OR TYPE CANCEL' After valid lib.sublib specifications are entered all COPYS statements are accepted and migrated like above under 'LIBDEF YES'. </pre>

CORGZ COPYS (CONTD.)

Conversion Proposal:

```
// EXEC LIBR
CONNECT SUBLIB=lib1.sublib1:lib2.sublib2
COPY name1.A [name2.E ...]          /* old COPYS A.name1 ... */
COPY *.A [*.E ... ]                 /* old COPYS A.ALL ... */
COPY *.A *.E [ ... ] REPLACE=YES /* old COPYS ALL --] specify-
    all source types contained in the from-SUBLIBRARY */
COPY *.A *.E [ ... ] REPLACE=NO /* old COPYS NEW --] specify-
    all source types contained in the from-SUBLIBRARY */
/*
```

Remarks:

REPLACE=NO is default - this fact has to be considered when converting to new COPY statements. The automatic migration uses REPLACE=YES. The list of copied members can be triggered by LIST=YES. The UPSI statement is no longer used.

CSERV DSPLY

Pre-VSE/Advanced Functions 2.1 Statement:	
(JCL statements: LIBDEF, ASSGN SYSCLB) <pre>// EXEC CSERV DSPLY name1[,name2, ...] DSPLY prog.ALL[,prog.ALL, ...] DSPLY ALL /*</pre>	
Behavior under VSE/Advanced Functions 2.1:	
(JCL statements LIBDEF, ASSGN SYSCLB see 'Job Control Functions'.) Operator action for failing JCL statements: Type EOB in order to continue processing. The CSERV migration program prompts for the missing lib.sublib information.	
LIBDEF YES:	<pre>DSPLY name1[,name2, ...] : accepted DSPLY prog.ALL[,prog.ALL, ...] : accepted DSPLY ALL : accepted</pre>
LIBDEF NO:	The first CSERV command causes the following prompting: <pre>'L023I LIBDEF MISSING FOR FROM - LIBRARY OF TYPE CL' 'L146A ENTER THE LIBRARY.SUBLIBRARY TO BE PROCESSED - OR TYPE CANCEL'</pre> After valid lib.sublib specification is entered all CSERV statements are accepted and migrated like the DSPLY statements above under 'LIBDEF YES'.

CSERV DSPLY (CONTD.)

Conversion Proposal:

```
// EXEC LIBR
ACCESS SUBLIB=lib.sublib
LIST name1.PHASE [name2.PHASE ...] /*old DSPLY name1 ... */
LIST prog*.PHASE [prog*.PHASE ...] /* old DSPLY prog.ALL ... */
LIST *.PHASE                      /* old DSPLY ALL */
/*
```

Remark:

Generic specification is generalized, it is no longer restricted to the first 4 characters of a phase name.

CSERV PUNCH

Pre-VSE/Advanced Functions 2.1 Statement:	
(JCL statements: LIBDEF, ASSGN SYSCLB) <pre>// EXEC CSERV PUNCH name1[,name2, ...] PUNCH prog.ALL[,prog.ALL, ...] PUNCH ALL /*</pre>	
Behavior under VSE/Advanced Functions 2.1:	
(JCL statements LIBDEF, ASSGN SYSCLB section 'Job Control Functions'.) Operator action for failing JCL statements: Type EOB in order to continue processing. The CSERV migration program prompts for the missing lib.sublib information.	
LIBDEF YES:	<pre>PUNCH name1[,name2, ...] : accepted PUNCH prog.ALL[,prog.ALL, ...] : accepted PUNCH ALL : accepted</pre>
LIBDEF NO:	The first CSERV command causes the following prompting: 'L023I LIBDEF MISSING FOR FROM - LIBRARY OF TYPE CL' 'L146A ENTER THE LIBRARY.SUBLIBRARY TO BE PROCESSED - OR TYPE CANCEL' After valid lib.sublib specification is entered all CSERV statements are accepted and migrated like the PUNCH statements above under 'LIBDEF YES'.

CSERV PUNCH (CONTD.)

Conversion Proposal:

```
// EXEC LIBR
ACCESS SUBLIB=lib.sublib
PUNCH name1.PHASE [name2.PHASE ...] EOF=NO
/* old PUNCH name1 ... */
PUNCH prog*.PHASE [prog*.PHASE ...] EOF=NO
/* old PUNCH prog.ALL ... */
PUNCH *.PHASE
/* old PUNCH ALL */
/*
```

Remark:

Generic specification is generalized, it is no longer restricted to the first 4 characters of a phase name.

CSERV DSPCH

Pre-VSE/Advanced Functions 2.1 Statement:	
(JCL statements: LIBDEF, ASSGN SYSCLB) <pre>// EXEC CSERV DSPCH name1[,name2, ...] DSPCH prog.ALL[,prog.ALL, ...] DSPCH ALL /*</pre>	
Behavior under VSE/Advanced Functions 2.1:	
(JCL statements LIBDEF, ASSGN SYSCLB see section 'Job Control Functions'.) Operator action for failing JCL statements: Type EOB in order to continue processing. The CSERV migration program prompts for the missing lib.sublib information.	
LIBDEF YES:	<pre>DSPCH name1[,name2, ...] : accepted DSPCH prog.ALL[,prog.ALL, ...] : accepted DSPCH ALL : accepted</pre>
LIBDEF NO:	The first CSERV command causes the following prompting: <pre>'L023I LIBDEF MISSING FOR FROM - LIBRARY OF TYPE CL' 'L146A ENTER THE LIBRARY.SUBLIBRARY TO BE PROCESSED - OR TYPE CANCEL'</pre> After valid lib.sublib specification is entered all CSERV statements are accepted and migrated like the DSPCH statements above under 'LIBDEF YES'.

CSERV DSPCH (CONTD.)

Conversion Proposal:

```
// EXEC LIBR
ACCESS SUBLIB=lib.sublib
LIST  name1.PHASE [name2.PHASE ...] /* old DSPCH name1 ... */
PUNCH name1.PHASE [name2.PHASE ...] EOF=NO
                                         /* old DSPCH name1 ... */
LIST  prog*.PHASE [prog*.PHASE ...] /* old DSPCH prog.ALL ... */
PUNCH prog*.PHASE [prog*.PHASE ...] EOF=NO
                                         /* old DSPCH prog.ALL ... */
LIST  *.PHASE                               /* old DSPCH ALL      */
PUNCH *.PHASE                               /* old DSPCH ALL      */
/*
```

Remark:

Generic specification is generalized, it is no longer restricted to the first 4 characters of a phase name.

PSERV DSPLY

Pre-VSE/Advanced Functions 2.1 Statement:	
(JCL statements: LIBDEF) <pre>// EXEC PSERV DSPLY name1[,name2, ...] DSPLY ALL /*</pre>	
Behavior under VSE/Advanced Functions 2.1:	
(JCL statements LIBDEF see section 'Job Control Functions'.) Operator action for failing JCL statements: Type EOB in order to continue processing. The PSERV migration program prompts for the missing lib.sublib information.	
LIBDEF YES:	DSPLY name1[,name2, ...] : accepted DSPLY ALL : accepted
LIBDEF NO:	The first PSERV command causes the following prompting: <pre>'L023I LIBDEF MISSING FOR FROM - LIBRARY OF TYPE PL' 'L146A ENTER THE LIBRARY.SUBLIBRARY TO BE PROCESSED - OR TYPE CANCEL'</pre> After valid lib.sublib specification is entered all PSERV statements are accepted and migrated like the DSPLY statements above under 'LIBDEF YES'.

PSERV DSPLY (CONTD.)

Conversion Proposal:

```
// EXEC LIBR
ACCESS SUBLIB=lib.sublib
LIST name1.PROC [name2.PROC ...] /*old DSPLY name1 ... */
LIST *.PROC /* old DSPLY ALL */
/*
```

PSERV PUNCH

Pre-VSE/Advanced Functions 2.1 Statement:	
(JCL statements: LIBDEF) // EXEC PSERV PUNCH name1[,name2, ...] PUNCH ALL /*	
Behavior under VSE/Advanced Functions 2.1:	
(JCL statements LIBDEF see section 'Job Control Functions'.) Operator action for failing JCL statements: Type EOB in order to continue processing. The PSERV migration program prompts for the missing lib.sublib information.	
LIBDEF YES:	PUNCH name1[,name2, ...] : accepted PUNCH ALL : accepted
LIBDEF NO:	The first PSERV command causes the following prompting: 'L023I LIBDEF MISSING FOR FROM - LIBRARY OF TYPE PL' 'L146A ENTER THE LIBRARY.SUBLIBRARY TO BE PROCESSED - OR TYPE CANCEL' After valid lib.sublib specification is entered all PSERV statements are accepted and migrated like the PUNCH statements above under 'LIBDEF YES'.

PSERV PUNCH (CONTD.)

Conversion Proposal:

```
// EXEC LIBR
ACCESS SUBLIB=lib.sublib1
PUNCH name1.PROC [name2.PROC ...] EOF=NO
/* old PUNCH name1 ... */
PUNCH *.PROC      /* old PUNCH ALL */
/*
```

PSERV DSPCH

Pre-VSE/Advanced Functions 2.1 Statement:	
(JCL statements: LIBDEF) <pre>// EXEC PSERV DSPCH name1[,name2, ...] DSPCH ALL /*</pre>	
Behavior under VSE/Advanced Functions 2.1:	
(JCL statements LIBDEF see section 'Job Control Functions'.) Operator action for failing JCL statements: Type EOB in order to continue processing. The PSERV migration program prompts for the missing lib.sublib information.	
LIBDEF YES:	DSPCH name1[,name2, ...] : accepted DSPCH ALL : accepted
LIBDEF NO:	The first PSERV command causes the following prompting: <pre>'L023I LIBDEF MISSING FOR FROM - LIBRARY OF TYPE PL' 'L146A ENTER THE LIBRARY.SUBLIBRARY TO BE PROCESSED - OR TYPE CANCEL'</pre> After valid lib.sublib specification is entered all PSERV statements are accepted and migrated like the DSPCH statements above under 'LIBDEF YES'.

PSERV DSPCH (CONTD.)

Conversion Proposal:

```
// EXEC LIBR
ACCESS SUBLIB=lib.sublib1
LIST  name1.PROC  [name2.PHASE ...] /* old DSPCH name1 ... */
PUNCH name1.PROC  [name2.PHASE ...] EOF=NO
                                           /* old DSPCH name1 ... */
LIST  *.PROC      /* old DSPCH ALL      */
PUNCH *.PROC      /* old DSPCH ALL      */
/*
```

RSERV DSPLY

Pre-VSE/Advanced Functions 2.1 Statement:	
(JCL statements: LIBDEF, ASSGN SYSRLB) <pre>// EXEC RSERV DSPLY name1[,name2, ...] DSPLY prog.ALL[,prog.ALL, ...] DSPLY ALL /*</pre>	
Behavior under VSE/Advanced Functions 2.1:	
(JCL statements LIBDEF, ASSGN SYSRLB see section 'Job Control Functions'.) Operator action for failing JCL statements: Type EOB in order to continue processing. The RSERV migration program prompts for the missing lib.sublib information.	
LIBDEF YES:	<pre>DSPLY name1[,name2, ...] : accepted DSPLY prog.ALL[,prog.ALL, ...] : accepted DSPLY ALL : accepted</pre>
LIBDEF NO:	The first RSERV command causes the following prompting: <pre>'L023I LIBDEF MISSING FOR FROM - LIBRARY OF TYPE RL' 'L146A ENTER THE LIBRARY.SUBLIBRARY TO BE PROCESSED - OR TYPE CANCEL'</pre> After valid lib.sublib specification is entered all RSERV statements are accepted and migrated like the DSPLY statements above under 'LIBDEF YES'.

RSERV DSPLY (CONTD.)

Conversion Proposal:

```
// EXEC LIBR
ACCESS SUBLIB=lib.sublib
LIST name1.OBJ [name2.OBJ ...] /*old DSPLY name1 ... */
LIST prog*.OBJ [prog*.OBJ ...] /* old DSPLY prog.ALL ... */
LIST *.OBJ /* old DSPLY ALL */
/*
```

Remark:

Generic specification is generalized, it is no longer restricted to the first 3 characters of an object deck name.

RSERV PUNCH

Pre-VSE/Advanced Functions 2.1 Statement:	
(JCL statements: LIBDEF, ASSGN SYSRLB) <pre>// EXEC RSERV PUNCH name1[,name2, ...] PUNCH prog.ALL[,prog.ALL, ...] PUNCH ALL /*</pre>	
Behavior under VSE/Advanced Functions 2.1:	
(JCL statements LIBDEF, ASSGN SYSRLB see section 'Job Control Functions'.) Operator action for failing JCL statements: Type EOB in order to continue processing. The RSERV migration program prompts for the missing lib.sublib information.	
LIBDEF YES:	<pre>PUNCH name1[,name2, ...] : accepted PUNCH prog.ALL[,prog.ALL, ...] : accepted PUNCH ALL : accepted</pre>
LIBDEF NO:	The first RSERV command causes the following prompting: <pre>'L023I LIBDEF MISSING FOR FROM - LIBRARY OF TYPE RL' 'L146A ENTER THE LIBRARY.SUBLIBRARY TO BE PROCESSED - OR TYPE CANCEL'</pre> After valid lib.sublib specification is entered all RSERV statements are accepted and migrated like the PUNCH statements above under 'LIBDEF YES'.

RSERV PUNCH (CONTD.)

Conversion Proposal:

```
// EXEC LIBR
  ACCESS SUBLIB=lib.sublib
  PUNCH name1.OBJ    [name2.OBJ    ...] EOF=NO
                                     /* old PUNCH name1 ... */
  PUNCH prog*.OBJ    [prog*.OBJ    ...] EOF=NO
                                     /* old PUNCH prog.ALL ... */
  PUNCH *.OBJ        /* old PUNCH ALL      */
/*
```

Remark:

Generic specification is generalized, it is no longer restricted to the first 3 characters of an object deck name.

RSERV DSPCH

Pre-VSE/Advanced Functions 2.1 Statement:	
(JCL statements: LIBDEF, ASSGN SYSRLB) <pre>// EXEC RSERV DSPCH name1[,name2, ...] DSPCH prog.ALL[,prog.ALL, ...] DSPCH ALL /*</pre>	
Behavior under VSE/Advanced Functions 2.1:	
(JCL statements LIBDEF, ASSGN SYSRLB see section 'Job Control Functions'.) Operator action for failing JCL statements: Type EOB in order to continue processing. The RSERV migration program prompts for the missing lib.sublib information.	
LIBDEF YES:	<pre>DSPCH name1[,name2, ...] : accepted DSPCH prog.ALL[,prog.ALL, ...] : accepted DSPCH ALL : accepted</pre>
LIBDEF NO:	The first RSERV command causes the following prompting: <pre>'L023I LIBDEF MISSING FOR FROM - LIBRARY OF TYPE RL' 'L146A ENTER THE LIBRARY.SUBLIBRARY TO BE PROCESSED - OR TYPE CANCEL'</pre> After valid lib.sublib specification is entered all RSERV statements are accepted and migrated like the DSPCH statements above under 'LIBDEF YES'.

RSERV DSPCH (CONTD.)

Conversion Proposal:

```
// EXEC LIBR
ACCESS SUBLIB=lib.sublib
LIST  name1.OBJ    [name2.OBJ    ...] /* old DSPCH name1 ... */
PUNCH name1.OBJ    [name2.OBJ    ...] EOF=NO
                                           /* old DSPCH name1 ... */
LIST  prog*.OBJ    [prog*.OBJ    ...] /* old DSPCH prog.ALL ... */
PUNCH prog*.OBJ    [prog*.OBJ    ...] EOF=NO
                                           /* old DSPCH prog.ALL ... */
LIST  *.OBJ        /* old DSPCH ALL      */
PUNCH *.OBJ        /* old DSPCH ALL      */
/*
```

Remark:

Generic specification is generalized, it is no longer restricted to the first 3 characters of an object deck name.

SSERV DSPLY

Pre-VSE/Advanced Functions 2.1 Statement:	
(JCL statements: LIBDEF, ASSGN SYSSLB) <pre>// EXEC SSERV DSPLY A.name1[,E.name2, ...] [,HEX] DSPLY A.ALL[,E.ALL, ...] [,HEX] DSPLY ALL[,HEX] /*</pre>	
Behavior under VSE/Advanced Functions 2.1:	
(JCL statements LIBDEF, ASSGN SYSSLB see section 'Job Control Functions'.) Operator action for failing JCL statements: Type EOB in order to continue processing. The SSERV migration program prompts for the missing lib.sublib information.	
LIBDEF YES:	<pre>DSPLY A.name1[,E.name2, ...][,HEX] : accepted DSPLY A.ALL[,E.ALL, ...][,HEX] : accepted DSPLY ALL[,HEX] : accepted</pre>
LIBDEF NO:	The first SSERV command causes the following prompting: <pre>'L023I LIBDEF MISSING FOR FROM - LIBRARY OF TYPE SL' 'L146A ENTER THE LIBRARY.SUBLIBRARY TO BE PROCESSED - OR TYPE CANCEL'</pre> After valid lib.sublib specification is entered all SSERV statements are accepted and migrated like the DSPLY statements above under 'LIBDEF YES'.

SSERV DSPLY (CONTD.)

Conversion Proposal:

```
// EXEC LIBR
ACCESS SUBLIB=lib.sublib
LIST name1.A [name2.E ...][FORMAT=HEX] /*old DSPLY A.name1 .. */
LIST *.A [*.E ...][FORMAT=HEX] /* old DSPLY A.ALL ... */
LIST *.A *.E [ ... ] [FORMAT=HEX] /* old DSPLY ALL .. specify -
                                all source types contained in the from-SUBLIBRARY */
/*
```

SSERV PUNCH

Pre-VSE/Advanced Functions 2.1 Statement:	
<pre> (JCL statements: LIBDEF, ASSGN SYSSLB) // EXEC SSERV PUNCH A.name1[,E.name2, ...] [,CMPRSD] PUNCH A.ALL[,E.ALL, ...] [,CMPRSD] PUNCH ALL[,CMPRSD] /* </pre>	
Behavior under VSE/Advanced Functions 2.1:	
(JCL statements LIBDEF, ASSGN SYSRLB see section 'Job Control Functions'.) Operator action for failing JCL statements: Type EOB in order to continue processing. The RSERV migration program prompts for the missing lib.sublib information.	
LIBDEF YES:	<pre> PUNCH A.name1[,E.name2, ...][,CMPRSD] : accepted PUNCH A.ALL[,E.ALL, ...][,CMPRSD] : accepted PUNCH ALL[,CMPRSD] : accepted </pre> Remark: Parameter 'CMPRSD' is ignored without message.
LIBDEF NO:	The first SSERV command causes the following prompting: <pre> 'L023I LIBDEF MISSING FOR FROM - LIBRARY OF TYPE SL' 'L146A ENTER THE LIBRARY.SUBLIBRARY TO BE PROCESSED - OR TYPE CANCEL' </pre> After valid lib.sublib specification is entered all SSERV statements are accepted and migrated like the PUNCH statements above under 'LIBDEF YES'.

SSERV PUNCH (CONTD.)

Conversion Proposal:

```
// EXEC LIBR
ACCESS SUBLIB=lib.sublib
PUNCH name1.A [name2.E ...] EOF=NO
                                /* old PUNCH A.name1 ... */
PUNCH *.A [*.E ...] EOF=NO
                                /* old PUNCH A.ALL ... */
PUNCH *.A *.E [ ... ]          /* old PUNCH ALL ... specify -
                                all source types contained in the from-SUBLIBRARY */
/*
```

Remark:

Punching of compressed members is no longer supported.

SSERV DSPCH

Pre-VSE/Advanced Functions 2.1 Statement:	
<pre> (JCL statements: LIBDEF, ASSGN SYSSLB) // EXEC SSERV DSPCH A.name1[,E.name2, ...] [,CMPRSD HEX CMPHEX] DSPCH A.ALL[,E.ALL, ...] [,CMPRSD HEX CMPHEX] DSPCH ALL[,CMPRSD HEX CMPHEX] /* </pre>	
Behavior under VSE/Advanced Functions 2.1:	
(JCL statements LIBDEF, ASSGN SYSSLB see section 'Job Control Functions'.) Operator action for failing JCL statements: Type EOB in order to continue processing. The SSERV migration program prompts for the missing lib.sublib information.	
LIBDEF YES:	<pre> DSPCH A.name1[,E.name2, ...] [,CMPRSD HEX CMPHEX] : accepted DSPCH A.ALL[,E.ALL, ...] [,CMPRSD HEX CMPHEX] : accepted DSPCH ALL[,CMPRSD HEX CMPHEX] : accepted </pre> Remarks: Parameter 'CMPRSD' is ignored without message. Parameter 'CMPHEX' : The HEX function is processed. The CMP function is ignored without message.
LIBDEF NO:	The first SSERV command causes the following prompting: 'L023I LIBDEF MISSING FOR FROM - LIBRARY OF TYPE SL' 'L146A ENTER THE LIBRARY.SUBLIBRARY TO BE PROCESSED - OR TYPE CANCEL' After valid lib.sublib specification is entered all SSERV statements are accepted and migrated like the DSPCH statements above under 'LIBDEF YES'.

SSERV DSPCH (CONTD.)

Conversion Proposal:

```
// EXEC LIBR
ACCESS SUBLIB=lib.sublib
LIST name1.A [name2.E ...][FORMAT=HEX] /* old DSPCH A.name1 . */
PUNCH name1.A [name2.E ...] EOF=NO
                                     /* old DSPCH A.name1 ... */
LIST *.A [*.E ...][FORMAT=HEX]      /* old DSPCH A.ALL ... */
PUNCH *.A [*.E ...] EOF=NO
                                     /* old DSPCH A.ALL ... */
LIST *.A *.E [ ... ] [FORMAT=HEX] /* old DSPCH ALL ... specify -
                                     all source types contained in the from-SUBLIBRARY */
PUNCH *.A *.E [ ... ]              /* old DSPCH ALL ... specify -
                                     all source types contained in the from-SUBLIBRARY */
/*
```

Remark:

Punching of compressed members is no longer supported.

DSERV

Pre-VSE/Advanced Functions 2.1 Statement:

(JCL statements: LIBDEF, ASSGN SYSCLB|SYSRLB|SYSSLB)

```
// EXEC DSERV
  DSPLY[S] CD[,...]
  DSPLY[S] CD(phasename[,nn])
  DSPLY[S] PD[,...]
  DSPLY[S] RD[,...]
  DSPLY[S] SD[,...]
  DSPLY[S] TD[,...]
  DSPLY[S] SDL[,...]
  DSPLY[S] ALL
/*

// EXEC DSERV                                displays status report only
/*
```

Behavior under VSE/Advanced Functions 2.1:

(JCL statements LIBDEF, ASSGN SYSCLB|SYSRLB|SYSSLB see
'Job Control Functions'.)

Operator action for failing JCL statements:

Type EOB in order to continue processing. The DSERV migration
program prompts for the missing lib.sublib information.

LIBDEF YES:	DSPLY[S] CD[,...]	accepted and migrated
	DSPLY[S] CD(phasename[,nn])	fails with message
	DSPLY[S] PD[,...]	accepted and migrated
	DSPLY[S] RD[,...]	accepted and migrated
	DSPLY[S] SD[,...]	accepted and migrated
	DSPLY[S] TD[,...]	accepted and migrated
	DSPLY[S] SDL[,...]	accepted and migrated
	DSPLY[S] ALL	accepted and migrated
	// EXEC DSERV	ignored without message
	/*	

DSERV (CONTD.)

LIBDEF NO:	DSPPLY[S] CD prompts for missing from-lib.sublib DSPPLY[S] PD prompts for missing from-lib.sublib DSPPLY[S] RD prompts for missing from-lib.sublib DSPPLY[S] SD prompts for missing from-lib.sublib DSPPLY[S] TD prompts for missing from-lib.sublib The prompting messages for DISPLY[S] CD and TD are: 'L023I LIBDEF MISSING FOR FROM - LIBRARY OF TYPE CL' 'L146A ENTER THE LIBRARY.SUBLIBRARY TO BE PROCESSED - OR TYPE CANCEL' The prompting messages for PD, RD, SD are the same with different types PL RL SL instead of CL. DSPPLY[S] CD(phasename[,nn]) fails with message 'L011I INVALID SYNTAX - - - (phasename[,nn])' and processing continues with next DSERV DSPPLY command if available. DSPPLY[S] ALL prompts for each missing LIBDEF in the range of the 4 types: CL, PL, RL, SL DSPPLY[S] SDL accepted and migrated without prompting. // EXEC DSERV ignored without message /*
---------------	--

Conversion Proposal:

```
// EXEC LIBR
  ACCESS lib.sublib
  LISTDIR *.PHASE                /* old DSPLY[S] CD */
  LISTDIR *.PROC                 /* old DSPLY[S] PD */
  LISTDIR *.OBJ                  /* old DSPLY[S] RD */
  LISTDIR *.A [*E ...]          /* old DSPLY[S] SD -
    specify all source types available in the sublibrary */
  LISTDIR $*.phase              /* old DSPLY[S] TD */
  LISTDIR SDL                   /* old DSPLY[S] SDL*/

  LISTDIR *.*                   /* old DSPLY[S] ALL*/
or
  LISTDIR lib.sublib1 [lib.sublib2 ...] /* old DSPLY[S] ALL*/

  LISTDIR SUBLIB=lib.sublib1 [lib.sublib2 ... ] OUTPUT=STATUS
    /* displays status reports for the specified sublibraries -
       can be used for DSERV steps without DSPLY statements */
/*
```

Remark:

Since VSE/Advanced Functions 2.1 directories are always sorted alphanumerically the LISTDIR output is always sorted, too.

MAINT CATALP

Pre-VSE/Advanced Functions 2.1 Statement:	
<pre>(JCL statements LIBDEF) // EXEC MAINT CATALP name[,VM=v.m][,EOP=yy][,DATA=NO YES] (member data) /*</pre>	
Behavior under VSE/Advanced Functions 2.1:	
(JCL statements LIBDEF see 'Job Control Functions'.) Operator action for failing JCL statements: Type EOB in order to continue processing. The MAINT migration program prompts for the missing lib.sublib information.	
LIBDEF YES:	Accepted and executed. The VM parameter is ignored without message. Cataloging is executed with REPLACE=YES. A '/+ JCL statement' is cataloged (MAINT CATALP only).
LIBDEF NO:	The first CATALP command causes the following prompting: 'L023I LIBDEF MISSING FOR TO - LIBRARY OF TYPE PL' 'L146A ENTER THE LIBRARY.SUBLIBRARY TO BE PROCESSED - OR TYPE CANCEL' After valid lib.sublib specification is entered all CATALP statements are accepted and migrated like the statement above under 'LIBDEF YES'.

Conversion Proposal:

```
// EXEC LIBR
  ACCESS SUBLIB=lib.sublib
  CATALOG name.PROC [EOD=yy] [DATA=NO|YES]
    ( member data )
/*
```

Remarks:

CATALP statements can also be processed by // EXEC LIBR. That means punched PSERV output can be processed directly by the program LIBR. 'v.m' is no longer supported - date / time stamps can be used for identification.

REPLACE=NO is default.

Information behind the End-of-Data indicator on the End-of Data statement is not stored in the library and therefore lost. A user may catalog a '/+ JCL statement' as last statement of a procedure by adding an End-of-Data statement with EOD other than '/+'.

In VSE/Advanced Functions 2.1 it is no longer necessary to specify a JCL command on the End-of Data statement ('/+') since the JCL JCL commands 'ASSGN SYSIN,...' and 'EXEC PROC=...' are accepted within a procedure.

MAINT CATALR

Pre-VSE/Advanced Functions 2.1 Statement:	
<pre> (JCL statements LIBDEF, ASSGN SYSRLB) // EXEC MAINT CATALR name[,v.m] (member data) /* </pre>	
Behavior under VSE/Advanced Functions 2.1:	
(JCL statements LIBDEF, ASSGN SYSRLB see 'Job Control Functions'.) Operator action for failing JCL statements: Type EOB in order to continue processing. The MAINT migration program prompts for the missing lib.sublib information.	
LIBDEF YES:	Accepted and executed. The vm parameter is ignored without message. Cataloging is executed with REPLACE=YES.
LIBDEF NO:	The first CATALR command causes the following prompting: 'L023I LIBDEF MISSING FOR TO - LIBRARY OF TYPE RL' 'L146A ENTER THE LIBRARY.SUBLIBRARY TO BE PROCESSED - OR TYPE CANCEL' After valid lib.sublib specification is entered all CATALR statements are accepted and migrated like the statement above under 'LIBDEF YES'.

MAINT CATALR (CONTD.)

Conversion Proposal:

```
// EXEC LIBR
  ACCESS SUBLIB=lib.sublib
  CATALOG name.OBJ
    ( member data )
/*
```

Remarks:

CATALR statements can also be processed by // EXEC LIBR. That means punched RSERV output can be processed directly by the program LIBR. 'v.m' is no longer supported - date / time stamps can be used for identification.
REPLACE=NO is default.

MAINT CATALS

Pre-VSE/Advanced Functions 2.1 Statement:	
<pre> (JCL statements LIBDEF, ASSGN SYSSLB) // EXEC MAINT CATALS A.name[,v.m[,C]] (member data containing BKEND or MACRO/MEND) /* </pre>	
Behavior under VSE/Advanced Functions 2.1:	
(JCL statements LIBDEF, ASSGN SYSSLB see 'Job Control Functions'.) Operator action for failing JCL statements: Type EOB in order to continue processing. The MAINT migration program prompts for the missing lib.sublib information.	
LIBDEF YES:	Accepted and executed. The v.m[,C] parameter is ignored without message. Cataloging is executed with REPLACE=YES. BKEND, MACRO, MEND statements are accepted. The parameters on the BKEND statement are ignored without message.
LIBDEF NO:	The first CATALS command causes the following prompting: 'L023I LIBDEF MISSING FOR TO - LIBRARY OF TYPE SL' 'L146A ENTER THE LIBRARY.SUBLIBRARY TO BE PROCESSED - OR TYPE CANCEL' After valid lib.sublib specification is entered all CATALS statements are accepted and migrated like for LIBDEF YES.

MAINT CATALS (CONTD.)

Conversion Proposal:

```
// EXEC LIBR
ACCESS SUBLIB=lib.sublib
CATALOG name.A
    ( member data - erase old BKEND statements,
                      close member by /+ or use EOD parameter )
/+
/*
```

Remarks:

CATALS statements can also be processed by // EXEC LIBR. That means punched SSERV output can be processed directly by the program LIBR. 'v.m[,C]' is no longer supported - date/time stamps can be used for identification.

REPLACE=NO is default.

MAINT DELETC

Pre-VSE/Advanced Functions 2.1 Statement:	
(JCL statements LIBDEF, ASSGN SYSCLB) <pre>// EXEC MAINT DELETC name1[,name2, ...] DELETC prog.ALL[,prog.ALL,...] DELETC ALL /*</pre>	
Behavior under VSE/Advanced Functions 2.1:	
(JCL statements LIBDEF, ASSGN SYSCLB see 'Job Control Functions'.) Operator action for failing JCL statements: Type EOB in order to continue processing. The MAINT migration program prompts for the missing lib.sublib information.	
LIBDEF YES:	Accepted and executed.
LIBDEF NO:	The first DELETC command causes the following prompting: <pre>'L023I LIBDEF MISSING FOR TO - LIBRARY OF TYPE CL' 'L146A ENTER THE LIBRARY.SUBLIBRARY TO BE PROCESSED - OR TYPE CANCEL'</pre> After valid lib.sublib specification is entered all DELETC statements are accepted and migrated like the statement above under 'LIBDEF YES'.

MAINT DELETC (CONTD.)

Conversion Proposal:

```
// EXEC LIBR
ACCESS SUBLIB=lib.sublib
DELETE name1.PHASE [name2.PHASE ... ] /* old DELETC name1,.. */
DELETE prog*.PHASE [prog*.PHASE ... ] /* old DELETC prog.ALL..*/
DELETE *.PHASE                          /* old DELETC ALL      */
/*
```

Remark:

Generic specification is generalized, it is no longer restricted to the first 4 characters of a phase name.

MAINT DELETEP

Pre-VSE/Advanced Functions 2.1 Statement:	
(JCL statements LIBDEF) <pre>// EXEC MAINT DELETEP name1[,name2, ...] DELETEP ALL /*</pre>	
Behavior under VSE/Advanced Functions 2.1:	
(JCL statements LIBDEF see 'Job Control Functions'.) Operator action for failing JCL statements: Type EOB in order to continue processing. The MAINT migration program prompts for the missing lib.sublib information.	
LIBDEF YES:	Accepted and executed.
LIBDEF NO:	The first DELETEP command causes the following prompting: <pre>'L023I LIBDEF MISSING FOR TO - LIBRARY OF TYPE PL' 'L146A ENTER THE LIBRARY.SUBLIBRARY TO BE PROCESSED - OR TYPE CANCEL'</pre> After valid lib.sublib specification is entered all DELETEP statements are accepted and migrated like the statement above under 'LIBDEF YES'.

MAINT DELETP (CONTD.)

Conversion Proposal:

```
// EXEC LIBR
ACCESS SUBLIB=lib.sublib
DELETE name1.PROC [name2.proc ... ] /* old DELETP name1,... */
DELETE *.PROC                       /* old DELETP ALL      */
/*
```

MAINT DELETR

Pre-VSE/Advanced Functions 2.1 Statement:	
(JCL statements LIBDEF, ASSGN SYSRLB) <pre>// EXEC MAINT DELETR name1[,name2, ...] DELETR prog.ALL[,prog.ALL,...] DELETR ALL /*</pre>	
Behavior under VSE/Advanced Functions 2.1:	
(JCL statements LIBDEF, ASSGN SYSRLB see 'Job Control Functions'.) Operator action for failing JCL statements: Type EOB in order to continue processing. The MAINT migration program prompts for the missing lib.sublib information.	
LIBDEF YES:	Accepted and executed.
LIBDEF NO:	The first DELETR command causes the following prompting: <pre>'L023I LIBDEF MISSING FOR TO - LIBRARY OF TYPE RL' 'L146A ENTER THE LIBRARY.SUBLIBRARY TO BE PROCESSED - OR TYPE CANCEL'</pre> After valid lib.sublib specification is entered all DELETR statements are accepted and migrated like the statement above under 'LIBDEF YES'.

MAINT DELETR (CONTD.)

Conversion Proposal:

```
// EXEC LIBR
ACCESS SUBLIB=lib.sublib
DELETE name1.OBJ [name2.OBJ ... ]    /* old DELETR name1,... */
DELETE prog*.OBJ [prog*.OBJ ... ]    /* old DELETR prog.ALL...*/
DELETE *.OBJ                          /* old DELETR ALL      */
/*
```

Remark:

Generic specification is generalized, it is no longer restricted to the first 3 characters of an object deck name.

MAINT DELETS

Pre-VSE/Advanced Functions 2.1 Statement:	
<pre> (JCL statements LIBDEF, ASSGN SYSSLB) // EXEC MAINT DELETS A.name1[,E.name2, ...] DELETS A.ALL DELETS ALL /* </pre>	
Behavior under VSE/Advanced Functions 2.1:	
(JCL statements LIBDEF, ASSGN SYSSLB see 'Job Control Functions'.) Operator action for failing JCL statements: Type EOB in order to continue processing. The MAINT migration program prompts for the missing lib.sublib information.	
LIBDEF YES:	Accepted and executed.
LIBDEF NO:	The first DELETS command causes the following prompting: <pre> 'L023I LIBDEF MISSING FOR TO - LIBRARY OF TYPE SL' 'L146A ENTER THE LIBRARY.SUBLIBRARY TO BE PROCESSED - OR TYPE CANCEL' </pre> After valid lib.sublib specification is entered all DELETS statements are accepted and migrated like the statement above under 'LIBDEF YES'.

MAINT DELETS (CONTD.)

Conversion Proposal:

```
// EXEC LIBR
ACCESS SUBLIB=lib.sublib
DELETE name1.A [name2.E ... ]      /* old DELETS A.name1,...*/
DELETE *.A                          /* old DELETS A.ALL...  */
DELETE *.A [*.E ....]             /* old DELETS ALL  - specify  -
                                all source types available in the sublibrary */
/*
```

MAINT RENAMC

Pre-VSE/Advanced Functions 2.1 Statement:	
(JCL statements LIBDEF, ASSGN SYSCLB) <pre>// EXEC MAINT RENAMC oldname1,newname1[,oldname2,newname2 ...] /*</pre>	
Behavior under VSE/Advanced Functions 2.1:	
(JCL statements LIBDEF, ASSGN SYSCLB see 'Job Control Functions'.) Operator action for failing JCL statements: Type EOB in order to continue processing. The MAINT migration program prompts for the missing lib.sublib information.	
LIBDEF YES:	Accepted and executed.
LIBDEF NO:	The first RENAMC command causes the following prompting: <pre>'L023I LIBDEF MISSING FOR TO - LIBRARY OF TYPE CL' 'L146A ENTER THE LIBRARY.SUBLIBRARY TO BE PROCESSED - OR TYPE CANCEL'</pre> After valid lib.sublib specification is entered all RENAMC statements are accepted and migrated like the statement above under 'LIBDEF YES'.

MAINT RENAMC (CONTD.)

Conversion Proposal:

```
// EXEC LIBR
  ACCESS SUBLIB=lib.sublib
  RENAME oldname1.PHASE:newname1.= [oldname2.PHASE:newname2.= .. ]
                                     /* old RENAMC oldname1,newname1, .. */
/*
```


MAINT RENAMP

Pre-VSE/Advanced Functions 2.1 Statement:	
(JCL statements LIBDEF) <pre>// EXEC MAINT RENAMP oldname1,newname1[,oldname2,newname2 ...] /*</pre>	
Behavior under VSE/Advanced Functions 2.1:	
(JCL statements LIBDEF see 'Job Control Functions'.) Operator action for failing JCL statements: Type EOB in order to continue processing. The MAINT migration program prompts for the missing lib.sublib information.	
LIBDEF YES:	Accepted and executed.
LIBDEF NO:	The first RENAMP command causes the following prompting: <pre>'L023I LIBDEF MISSING FOR TO - LIBRARY OF TYPE PL' 'L146A ENTER THE LIBRARY.SUBLIBRARY TO BE PROCESSED - OR TYPE CANCEL'</pre> After valid lib.sublib specification is entered all RENAMP statements are accepted and migrated like the statement above under 'LIBDEF YES'.

MAINT RENAMP (CONTD.)

Conversion Proposal:

```
// EXEC LIBR
  ACCESS SUBLIB=lib.sublib
  RENAME oldname1.PROC:newname1.= [oldname2.PROC:newname2.= ... ]
                                     /* old RENAMP oldname1,newname1, ... */
/*
```

MAINT RENAMR

Pre-VSE/Advanced Functions 2.1 Statement:	
<pre>(JCL statements LIBDEF, ASSGN SYSRLB) // EXEC MAINT RENAMR oldname1,newname1[,oldname2,newname2 ...] /*</pre>	
Behavior under VSE/Advanced Functions 2.1:	
(JCL statements LIBDEF, ASSGN SYSRLB see 'Job Control Functions'.) Operator action for failing JCL statements: Type EOB in order to continue processing. The MAINT migration program prompts for the missing lib.sublib information.	
LIBDEF YES:	Accepted and executed.
LIBDEF NO:	The first RENAMR command causes the following prompting: 'L023I LIBDEF MISSING FOR TO - LIBRARY OF TYPE RL' 'L146A ENTER THE LIBRARY.SUBLIBRARY TO BE PROCESSED - OR TYPE CANCEL' After valid lib.sublib specification is entered all RENAMR statements are accepted and migrated like the statement above under 'LIBDEF YES'.

MAINT RENAMR (CONTD.)

Conversion Proposal:

```
// EXEC LIBR
ACCESS SUBLIB=lib.sublib
RENAME oldname1.OBJ:newname1.= [oldname2.OBJ:newname2.= ... ]
/* old RENAMR oldname1,newname1, ... */
/*
```

MAINT RENAMS

Pre-VSE/Advanced Functions 2.1 Statement:	
<pre> (JCL statements LIBDEF, ASSGN SYSSLB) // EXEC MAINT RENAMS A.oldname1,A.newname1[,E.oldname2,E.newname2, ...] /* </pre>	
Behavior under VSE/Advanced Functions 2.1:	
(JCL statements LIBDEF, ASSGN SYSSLB see 'Job Control Functions'.) Operator action for failing JCL statements: Type EOB in order to continue processing. The MAINT migration program prompts for the missing lib.sublib information.	
LIBDEF YES:	Accepted and executed.
LIBDEF NO:	The first RENAMS command causes the following prompting: <pre> 'L023I LIBDEF MISSING FOR TO - LIBRARY OF TYPE SL' 'L146A ENTER THE LIBRARY.SUBLIBRARY TO BE PROCESSED - OR TYPE CANCEL' </pre> After valid lib.sublib specification is entered all RENAMS statements are accepted and migrated like the statement above under 'LIBDEF YES'.

MAINT RENAMS (CONTD.)

Conversion Proposal:

```
// EXEC LIBR
ACCESS SUBLIB=lib.sublib
RENAME oldname1.A:newname1.= [oldname2.E:newname2.= ... ]
                                /* old RENAMS A.oldname1,A.newname1, ... */
/*
```

MAINT CONDS

Pre-VSE/Advanced Functions 2.1 Statement:

```
(JCL statements LIBDEF, ASSGN SYSCLB|SYSRLB|SYSSLB)

// EXEC MAINT
   CONDS CL|PL|RL|SL
/*
```

Behavior under VSE/Advanced Functions 2.1:

(JCL statements LIBDEF, ASSGN SYSCLB|SYSRLB|SYSSLB see
'Job Control Functions'.)

Operator action for failing JCL statements:

Type EOB in order to continue processing.

The CONDS command is ignored without message. Processing continues with next MAINT command if available.

Conversion Proposal:

Not applicable, since a Condense function is no longer needed with the new library structure.

MAINT CONDL

Pre-VSE/Advanced Functions 2.1 Statement:
(JCL statements LIBDEF, ASSGN SYSCLB SYSRLB SYSSLB) // EXEC MAINT CONDL lib=n /*
Behavior under VSE/Advanced Functions 2.1:
(JCL statements LIBDEF, ASSGN SYSCLB SYSRLB SYSSLB see 'Job Control Functions'.) Operator action for failing JCL statements: Type EOB in order to continue processing. The CONDS command is ignored without message. Processing continues with next MAINT command if available.
Conversion Proposal:
Not applicable, since a Condense-limit function is no longer needed with the new library structure.

CHAPTER 6. USER FILES TRANSFER

INTRODUCTION

You can migrate the following files using the VSE/VSAM Backup/Restore Feature:

- Native VSAM files
- DL/I VSAM files
- SQL/ VSAM files

Define the VSAM catalog and space before you restore them to VSE/SP 2.1.

Define a log data set for SQL/DS and format it via a SQL **coldlog** startup before the normal startup if the database is not restored from the SQL ARCHIVE copy.

Samples for the SQL startup are listed under "SQL Initialization" on page 314.

MIGRATE DATABASES

```

* $$ JOB JNM=BACKUP,DISP=D,CLASS=0,PRI=3
* $$ LST CLASS=R
// JOB BACKUP ALL
* *****
*
*   BACKUP ALL VSAM OBJECTS CONTAINED IN THE USER CATALOG
*   VSESP.SYSWK2.USER.CATALOG ON A SIPO 1.4 SYSTEM
*   EXCEPT FOR SOME THAT DO NOT NEED TO BE MIGRATED.
*
*   THE BACKUP INCLUDES
*   THE DL/I DATABASE WITH GENERIC DATASET NAMES SAMPLE.*
*   AND THE SQL DATABASE WITH GENERIC DATASET NAMES SQL.*
*
* *****
* *****
*   PLEASE MOUNT BACKUP TAPE ON 1A1
* *****
// PAUSE
// ASSGN SYS005,1A1
// MTC REW,SYS005
// TLBL BACKUP,'BACKUP'
// DLBL IJSYSUC,'VSESP.SYSWK2.USER.CATALOG',,VSAM
// EXEC IDCAMS,SIZE=AUTO
  BACKUP (*)
      STDLABEL(BACKUP)
      EXCLUDE (FORREC.* TEST.* TSTDREC.*)
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=VSAMDEFU,DISP=D,CLASS=0,PRI=3
* $$ LST CLASS=R,DEST=(*,KETTNER)
* *****
*      DEFINE USER CATALOG AND VSAM SPACE IN THE VSE/SP 2.1 SYSTEM      *
* *****
// JOB VSAMDEFU USERCATALOG AND SPACE
// EXEC IDCAMS,SIZE=AUTO
  DEFINE USERCATALOG
    ( NAME(VSESP2.SYSWK3.USER.CATALOG),
      ORIGIN(187674),
      BLOCKS(744),
      VOLUME(SYSWK3))
  DEFINE SPACE
    ( ORIGIN(188418),
      BLOCKS(199950),
      VOLUMES(SYSWK3))
    CATALOG(VSESP2.SYSWK3.USER.CATALOG)
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=VSAMDEFL,DISP=D,CLASS=0,PRI=3
* $$ LST CLASS=R,DEST=(*,KETTNER)
* *****
*
*      THIS JOB IS REQUIRED IF THE SQL DATABASE IS TO BE
*      RESTORED VIA THE VSAM BACKUP/RESTORE FEATURE
*
*
*      DEFINE A LOG DATASET FOR SQL/DS
*
* *****
// JOB VSAMDEFL NEW SQL LOG DATASET
// EXEC IDCAMS,SIZE=AUTO
  DEFINE CLUSTER
    ( NAME(SQL.LOGDSK1.STARTER.DB),
      CONTROLINTERVALSIZE(4096)
      BLOCKS(10416 0)
      NONINDEXED
      VOLUMES(SYSWK3)
      RECORDSIZE(4089 4089)
      REUSE
      SHAREOPTIONS(1) )
    CATALOG(VSESP2.SYSWK3.USER.CATALOG)
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=VSAMDEFS,DISP=D,CLASS=0,PRI=3
* $$ LST CLASS=R,DEST=(*,KETTNER)
* ****
*
*   THIS JOB IS REQUIRED IF A SQL ARCHIVE COPY OF THE
*   DATABASE IS TO BE RESTORED
*
*   DEFINE SQL/DS VSAM DATASETS
*
* ****
// JOB VSAMDEFS NEW SQL DATASETS
// EXEC IDCAMS,SIZE=AUTO
  DEFINE CLUSTER
    ( NAME(SQL.LOGDSK1.STARTER.DB)
      CONTROLINTERVALSIZE(4096)
      BLOCKS(10416 0)
      NONINDEXED
      VOLUMES(SYSWK3)
      RECORDSIZE(4089 4089)
      REUSE
      SHAREOPTIONS(1) )
    CATALOG(VSESP2.SYSWK3.USER.CATALOG)
  DEFINE CLUSTER
    ( NAME(SQL.BDISK.STARTER.DB)
      CONTROLINTERVALSIZE(512)
      BLOCKS(24552 0)
      NONINDEXED
      VOLUMES(SYSWK3)
      RECORDSIZE(505 505)
      REUSE
      SHAREOPTIONS(1) )
    CATALOG(VSESP2.SYSWK3.USER.CATALOG)
  DEFINE CLUSTER
    ( NAME(SQL.DDSK1.STARTER.DB)
      CONTROLINTERVALSIZE(4096)
      BLOCKS(120528 0)
      NONINDEXED
      VOLUMES(SYSWK3)
      RECORDSIZE(4089 4089)
      REUSE
      SHAREOPTIONS(1) )
    CATALOG(VSESP2.SYSWK3.USER.CATALOG)
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=RESTDLI,DISP=D,CLASS=0,PRI=3
* $$ LST CLASS=R,DEST=(*,KETTNER)
// JOB RESTDLI DL/I DATASETS
* *****
*
*   RESTORE DL/I DATASETS BACKED UP FROM THE
*   VSESP.SYSWK2.USER.CATALOG ON A SIPO 1.4 SYSTEM
*
*   THE DL/I DATASETS HAVE A GENERIC NAME OF SAMPLE.
*
* *****
* *****
*   PLEASE MOUNT BACKUP TAPE ON 1B2
* *****
// PAUSE
// ASSGN SYS004,1B2
// MTC REW,SYS004
// TLBL RESTORE,'BACKUP'
// DLBL IJSYSUC,'VSESP2.SYSWK3.USER.CATALOG',,VSAM
// EXEC IDCAMS,SIZE=AUTO
  RESTORE OBJECTS
    (SAMPLE.*)
    VOLUMES (SYSWK3)
    STDLABEL(RESTORE)
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=RESTSQL,DISP=D,CLASS=0,PRI=3
* $$ LST CLASS=R,DEST=(*,KETTNER)
// JOB RESTSQL SQL/DS DATASETS
* *****
*
*   RESTORE SQL/DS VSAM DATASETS BACKED UP FROM THE
*   VSESP.SYSWK2.USER.CATALOG ON A SIPO 1.4 SYSTEM
*   THE LOG DATASET IS EXCLUDED.
*
* *****
* *****
*   PLEASE MOUNT BACKUP TAPE ON 1B2
* *****
// PAUSE
// ASSGN SYS004,1B2
// MTC REW,SYS004
// TLBL RESTORE,'BACKUP'
// DLBL IJSYSUC,'VSESP2.SYSWK3.USER.CATALOG',,VSAM
// EXEC IDCAMS,SIZE=AUTO
  RESTORE OBJECTS
      (SQL.*)
      EXCLUDE (SQL.LOGDSK1.STARTER.DB)
      VOLUMES (SYSWK3)
      STDLABEL(RESTORE)
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=RESTORE,DISP=D,CLASS=0,PRI=3
* $$ LST CLASS=R,DEST=(*,KETTNER)
// JOB RESTORE VSAM DATASETS
* ****
*
* RESTORE NATIVE VSAM DATASETS BACKED UP FROM THE
* VSESP.SYSWK2.USER.CATALOG ON A SIPO 1.4 SYSTEM
*
* ONLY THE FILES WITH A GENERIC NAME OF MIGR.* ARE RESTORED.
*
* ****
* ****
* PLEASE MOUNT BACKUP TAPE ON 1B2
* ****
// PAUSE
// ASSGN SYS004,1B2
// MTC REW,SYS004
// TLBL RESTORE,'BACKUP'
// DLBL IJSYSUC,'VSESP2.SYSWK3.USER.CATALOG',,VSAM
// EXEC IDCAMS,SIZE=AUTO
  RESTORE OBJECTS
    (MIGR.*)
    VOLUMES (SYSWK3)
    STDLABEL(RESTORE)
/*
/&
* $$ EOJ

```

SQL INITIALIZATION

```
* $$ JOB JNM=SQLSTART,DISP=D,CLASS=5,PRI=3
* $$ LST CLASS=R,DEST=(*,KETTNER)
* ****
*
*   THIS IS A SAMPLE OF A NORMAL SQL/DS STARTUP
*
*   SQL STARTUP AFTER DATABASE RESTORE
*
* ****
// JOB SQLSTART
// DLBL IJSYSUC,'VSESP2.SYSWK3.USER.CATALOG',,VSAM
// DLBL BDISK,'SQL.BDISK.STARTER.DB',,VSAM
// DLBL LOGDSK1,'SQL.LOGDSK1.STARTER.DB',,VSAM
// DLBL DDSK1,'SQL.DDSK1.STARTER.DB',,VSAM
// LIBDEF *,SEARCH=PRD2.DBASE
// TLBL ARIARCH,'SQLARC'
// EXEC ARISQLDS,SIZE=AUTO
/*
/&
* $$ EOJ
```

```
* $$ JOB JNM=SQLST21C,DISP=D,CLASS=5,PRI=3
* $$ LST CLASS=R,DEST=(*,KETTNER)
* ****
*
*   THIS JOB HAS TO BE RUN BEFORE THE NORMAL STARTUP IF VSAM
*   BACKUP/RESTORE WAS USED. IT FORMATS THE LOG DATASET
*
*   PERFORM AN SQL COLDLOG TO FORMAT THE LOG DATASET
*
* ****
// JOB SQLST21C
// DLBL IJSYSUC,'VSESP2.SYSWK3.USER.CATALOG',,VSAM
// DLBL BDISK,'SQL.BDISK.STARTER.DB',,VSAM
// DLBL LOGDSK1,'SQL.LOGDSK1.STARTER.DB',,VSAM
// DLBL DDSK1,'SQL.DDSK1.STARTER.DB',,VSAM
// LIBDEF *,SEARCH=PRD2.DBASE
// EXEC ARISQLDS,SIZE=AUTO,PARM='STARTUP=L,SYSMODE=S,DUALLOG=N'
/*
/&
* $$ EOJ
```



```

* $$ JOB JNM=SQLST21R,DISP=D,CLASS=5,PRI=3
* $$ LST CLASS=R,DEST=(*,KETTNER)
* ****
*
*   THIS JOB HAS TO BE RUN BEFORE A NORMAL STARTUP IF AN SQL
*   ARCHIVE COPY OF THE DATABASE HAS TO BE RESTORED
*
*   START SQL/DS TO RESTORE ARCHIVED COPY OF THE DATABASES
*
* ****
// JOB SQLST21R
// DLBL IJSYSUC,'VSESP2.SYSWK3.USER.CATALOG',,VSAM
// DLBL BDISK,'SQL.BDISK.STARTER.DB',,VSAM
// DLBL LOGDSK1,'SQL.LOGDSK1.STARTER.DB',,VSAM
// DLBL DDSK1,'SQL.DDSK1.STARTER.DB',,VSAM
// LIBDEF *,SEARCH=PRD2.DBASE
// TLBL ARIARCH,'SQLARC'
// EXEC ARISQLDS,SIZE=AUTO,PARM='STARTUP=R,LOGMODE=A'
/*
/&
* $$ EOJ

```


CHAPTER 7. ACF/VTAM CONSIDERATIONS

VTAM BOOKS

The following are IBM-supplied VTAM Books. The changes which have been made since VSE/SP 1.1.0 are shown at the end of each book.

VTAM STARTUP BOOK

CATALOG ATCSTR00.B REPLACE=YES

SSCPID=1,
HOSTSA=1,
MAXSUBA=63,
MAXAPPL=30,
CONFIG=00,
LIST=00,
NOPROMPT,
IOINT=0,
LFBUF=(70,288,,6,),
LPBUF=(54,,6),
SFBUF=(153,,20),
SPBUF=(210,,20),
VFBUF=28000,
VPBUF=500000,
WPBUF=(100,,10),
VTAMEAS=64

/+

C
C
C
C
C
C
C
C
C
C
C
C
C
C
C

Explanations are given on the following page.

- MAXSUBA=63 **was** MAXSUBA=31 in VSE/SP 1.1.0
- MAXAPPL=30 **was** MAXAPPL=20 in VSE/SP 1.1.0
- CONFIG=00,LIST=00,NOPROMPT **was** CONFIG=33, LIST=33,PROMPT in VSE/SP 1.1.0.

The normal VTAM start job is supplied without PROMPT option.

If you want to use the PROMPT option you can add it in member 'ATCSTR00.B'.

Use of the PROMPT option allows you to change the network, or other option lists, at startup time.

VSE/SP 1.1.0 used the PROMPT option to allow specification of the terminal startup list (List=33) and (Config=33). In VSE/SP 2.1, the supplied ATCSTR00.B supports addresses in the PRD2.CONFIG library.

- IOINT=0 **was** IOINT=180 in VSE/SP 1.1.0
- LFBUF=(70,288,,6,) **was** LFBUF=(68,128,,5,) in VSE/SP 1.1.0
- VFBUF=28000,VPBUF=500000 **was** VFBUF=16384,VPBUF=262144 in VSE/SP 1.1.0
- VTAMEAS=64 **was** VTAMEAS=404 in VSE/SP 1.1.0
- Default parameters which are not stated explicitly in ATCSTR00.B

CDRSCTI=480, COLD, ITLIM=0, MSGMOD=NO, SONLIM=(60,30),
SUPP=NOSUP, NOTNSTAT, NOTRACE, TRACE, TYPE=VTAM, MODE=INT,
OPTION=(API,MSG,PIU), SIZE=2, USSTAB

The ACF/VTAM 2.1 Planning and Installation Reference provides a detailed description of these parameters.

Configuration List

CATALOG ATCCON00.B REPLACE=YES

VTMAPPL,
VTMSNA,
VTMNSNA,
VTMCA1,
VTMCA2,
VTMCA3,
VTMPATH,
VTMCDRM,
VTMCDRS,
VTMSW1

C
C
C
C
C
C
C
C
C

/+

- VTMAPPL : for definition of VTAM applications.

IESWAITT is a VTAM application which is used by other partitions to check that VTAM is started. So for example you will see in the VSE/ICCF partition startup a statement **// EXEC IESWAITT**. This program will wait until the VTAM appl-id IESWAITT is started before allowing the POWER startup to proceed.

When the VTAM startup is successful, the normal status as shown by the command **D NET,APPLS** for these applications (PNET and IESWAITT) is **CONCT**.

- VTMSNA : for definition of SNA local terminals
- VTMNSNA : for definition of non-SNA local terminals
- VTMCA1 : member for ICA lines to remote terminals
- VTMCA2 : member for ICA lines to other systems
- VTMCA3 : member for ICA switched (dial) connections
- VTMPATH : for routing definitions (explicit route,virtual route)
- VTMCDRM : member for cross domain resource managers
- VTMCDRS : member for cross domain resources
- VTMSW1 : member for switched network resources

Other Books

CATALOG VTMAPPL.B REPLACE=YES

```
VTMAPPL  VBUILD TYPE=APPL
DBDCCICS APPL  AUTH=(PASS,ACQ)
POWER    APPL  AUTH=(ACQ)
PNET     APPL  AUTH=(PASS,ACQ),VPACING=3,MODETAB=VTMLOGTB,DLOGMOD=PNET
IESWAITT APPL  AUTH=(NOACQ)
/+
```

Performance options: VPACING=3, MODETAB=VTMLOGTB

CATALOG VTMSNA.B REPLACE=YES

```
VTMSNA  VBUILD TYPE=LOCAL
DP108C01 PU    CUADDR=08C,                                     C
                PUTYPE=2, ISTATUS=ACTIVE, MAXBFRU=(1,2)
L0108C01 LU    LOCADDR=2,                                     C
                SSCPFM=USSSCS,                                 C
                USSTAB=VTMUSSTR, PACING=1, VPACING=2,
                ISTATUS=ACTIVE, LOGAPPL=DBDCCICS
L1108C01 LU    LOCADDR=3,                                     C
                SSCPFM=USSSCS, USSTAB=VTMUSSTR, PACING=1, VPACING=2, C
                ISTATUS=ACTIVE, LOGAPPL=DBDCCICS
L2108C01 LU    LOCADDR=4,                                     C
                SSCPFM=USSSCS, USSTAB=VTMUSSTR, PACING=1, VPACING=2, C
                ISTATUS=ACTIVE, LOGAPPL=DBDCCICS
L3108C01 LU    LOCADDR=5,                                     C
                SSCPFM=USSSCS, USSTAB=VTMUSSTR, PACING=1, VPACING=2, C
                ISTATUS=ACTIVE, LOGAPPL=DBDCCICS
L4108C01 LU    LOCADDR=6,                                     C
                SSCPFM=USSSCS, USSTAB=VTMUSSTR, PACING=1, VPACING=2, C
                ISTATUS=ACTIVE, LOGAPPL=DBDCCICS
L5108C01 LU    LOCADDR=7,                                     C
                SSCPFM=USSSCS, USSTAB=VTMUSSTR, PACING=1, VPACING=2, C
                ISTATUS=ACTIVE, LOGAPPL=DBDCCICS
/+
```

Performance options: MAXBFRU=(1,2), USSTAB=VTMUSSTR, PACING=1,
VPACING=2

CATALOG VTMNSNA.B REPLACE=YES

VTMNSNA LBUILD

D080	LOCAL CUADDR=080,TERM=3277,FEATUR2=(MODEL2), USSTAB=VTMUSSTB,DLOGMOD=S3270,LOGAPPL=DBDCCICS	C
D081	LOCAL CUADDR=081,TERM=3277,FEATUR2=(MODEL2), USSTAB=VTMUSSTB,DLOGMOD=S3270,LOGAPPL=DBDCCICS	C
D082	LOCAL CUADDR=082,TERM=3277,FEATUR2=(MODEL2), USSTAB=VTMUSSTB,DLOGMOD=S3270,LOGAPPL=DBDCCICS	C
D083	LOCAL CUADDR=083,TERM=3277,FEATUR2=(MODEL2), USSTAB=VTMUSSTB,DLOGMOD=S3270,LOGAPPL=DBDCCICS	C
D084	LOCAL CUADDR=084,TERM=3277,FEATUR2=(MODEL2), USSTAB=VTMUSSTB,DLOGMOD=S3270,LOGAPPL=DBDCCICS	C
D085	LOCAL CUADDR=085,TERM=3277,FEATUR2=(MODEL2), USSTAB=VTMUSSTB,DLOGMOD=S3270,LOGAPPL=DBDCCICS	C
D086	LOCAL CUADDR=086,TERM=3277,FEATUR2=(MODEL2), USSTAB=VTMUSSTB,DLOGMOD=S3270,LOGAPPL=DBDCCICS	C
D087	LOCAL CUADDR=087,TERM=3277,FEATUR2=(MODEL2), USSTAB=VTMUSSTB,DLOGMOD=S3270,LOGAPPL=DBDCCICS	C
D088	LOCAL CUADDR=088,TERM=3277,FEATUR2=(MODEL2), USSTAB=VTMUSSTB,DLOGMOD=S3270,LOGAPPL=DBDCCICS	C
D089	LOCAL CUADDR=089,TERM=3277,FEATUR2=(MODEL2), USSTAB=VTMUSSTB,DLOGMOD=S3270,LOGAPPL=DBDCCICS	C
D08A	LOCAL CUADDR=08A,TERM=3277,FEATUR2=(MODEL2), USSTAB=VTMUSSTB,DLOGMOD=S3270,LOGAPPL=DBDCCICS	C
D08B	LOCAL CUADDR=08B,TERM=3277,FEATUR2=(MODEL2), USSTAB=VTMUSSTB,DLOGMOD=S3270,LOGAPPL=DBDCCICS	C

/+

CHAPTER 8. IBM PRODUCTS TRANSFER

MIGRATE AN OLD PRODUCT

The following pages show suggestions for how to migrate a product together with its history information. A brief description of each sample job is listed below:

MIGRATE0 Backs up a complete library set from the pre-VSE/SP 2.1 system to tape together with the system history file.

MIGRATE1 Creates, on the VSE/SP 2.1 system, a temporary history file and a new library PRD3.

MIGRATE2 Using MSHP, install the product from the backup tape into PRD3.OPTIONAL using a temporary history file.

MIGRATE3 Again using MSHP, run a retrace against the temporary history file.

A sample of this printout follows the job control. Note that the product CB1E44 (new product number) comprises 5746-CB100 (old number). When referring to this product with MSHP under VSE/SP 2.1, the new number should be used.

MIGRATE4 This job defines a new sublibrary, PRD3.COBOl. Then, having CONNECTed this new library to PRD3.OPTIONAL, it moves all COBOl modules into it.

MIGRATE5 This job sets up residence for the product CB1E44 in sublibrary PRD3.COBOl. Then it backs up the sublibrary together with the history for that product.

Note: Only this product should be in the sublibrary.

MIGRATE6 This job defines a new sublibrary PRD3.NEWCOb and restores the product CB1E44 into it with the history information for that product merged into the system history file.

```

* $$ JOB JNM=MIGRATE0,CLASS=A,DISP=D,PRI=3
* $$ LST CLASS=Q,DISP=H
// JOB MIGRATE0 BACKUP LIBRARY SET AND SYSTEM HISTORY FILE ON OLD SYSTEM
// DLBL IJSYSCL,'PROD.CORE.IMAGE.LIBRARY.C'
// EXTENT ,SYSWK1
// DLBL IJSYSRL,'DOS/VSE.GEN.SYSRLB.C'
// EXTENT ,DOSRES
// DLBL IJSYSSL,'DOS/VSE.GEN.SYSSLB.C'
// EXTENT ,DOSRES
// PAUSE MOUNT TAPE ON 1B0
// ASSGN SYS006,1B0
// MTC REW,SYS006
// EXEC BACKUP
NOSA
BACKUP CL,LIB=IJSYSCL
BACKUP RL,LIB=IJSYSRL
BACKUP SL,LIB=IJSYSSL
/*
// MTC RUN,1B0
/&
* $$ EOJ

```

```

* $$ JOB JNM=MIGRATE1,CLASS=A,DISP=H,PRI=3,NTFY=YES
* $$ LST CLASS=Q,DISP=H
// JOB MIGRATE1 CREATE TEMP HISTORY FILE AND LIBRARY
// DLBL IJSYSHF,'TEMP.OLD.HIST.FILE'
// EXTENT SYS018,SYSWK2,1,0,204000,3460
// ASSGN SYS018,242
// EXEC MSHP
CREATE HIST SYS
/*
// DLBL PRD3,'TEMP.OLD.LIB'
// EXTENT ,SYSWK2,1,0,4000,200000
// EXEC LIBR
DEFINE LIB=PRD3
/*
/&
* $$ EOJ

```

*** \$\$ JOB JNM=MIGRATE2,CLASS=A,DISP=H,PRI=3,NTFY=YES**

```
// JOB MIGRATE2 INSTALL OLD LIBRARY SET
// LIBDEF PHASE,SEARCH=(IJSYSRS.SYSLIB,PRD1.BASE)
// PAUSE MOUNT TAPE CONTAINING OLD LIBRARY SET
// ASSGN SYS006,1B1
// MTC REW,SYS006
// DLBL IJSYSHF,'TEMP.OLD.HIST.FILE'
// EXTENT SYS018,SYSWK2,1,0,204000,3460
// ASSGN SYS018,242
// DLBL PRD3,'TEMP.OLD.LIB'
// EXTENT ,SYSWK2,1,0,4000,200000
// EXEC MSHP
INSTALL      PRODUCT FROMT PROD IN=PRD3.OPTIONAL -
      GENE IN=PRD3.OPTIONAL
/*
/&
* $$ EOJ
```

*** \$\$ JOB JNM=MIGRATE3,CLASS=A,DISP=H,PRI=3,NTFY=YES**

```
// JOB MIGRATE3 RETRACE TEMP HIST FILE TO ASCERTAIN NEW PRODUCT NUMBER
// LIBDEF PHASE,SEARCH=(IJSYSRS.SYSLIB,PRD1.BASE)
// DLBL IJSYSHF, 'TEMP.OLD.HIST.FILE'
// EXTENT SYS018,SYSWK2,1,0,204000,3460
// ASSGN SYS018,242
// EXEC MSHP
RETRACE      PRODUCT
/*
/&
* $$ EOJ
```

Sample Output of RETRACE PRODUCT

```
// JOB MIGRATE3 RETRACE TEMP HIST FILE TO ASCERTAIN NEW PRODUCT NUMBER
// LIBDEF PHASE,SEARCH=(IJSYSRS.SYSLIB,PRD1.BASE)
// DLBL IJSYSHF,'TEMP.OLD.HIST.FILE'
// EXTENT SYS018,SYSWK2,1,0,204000,3460
// ASSGN SYS018,242
// EXEC MSHP
RETRACE PRODUCT
M039I RETRACE IN PROGRESS
DATE 09/07/84 (MM/DD/YY) JOB RETRACE MSHP-FUNCTION = RETRACE PRODUCT
                        CUSTOMER      ABC DEPT
                        ADDRESS        Main Street
                        PHONE NUMBER   123 - 456 7890
                        SYSTEM PROGRAMMER JOHN
                        ENVIRONMENT     VSE/SP 2.1
                        HISTORY FILE    19 % IN USE
DATE 09/07/84 (MM/DD/YY) JOB RETRACE MSHP-FUNCTION = RETRACE PRODUCT
                        PRODUCT LIST
                        .
                        .
                        .
                        .
                        .
PRODUCT CB1E44 INSTALLATION DATE=-07/09/84
                        COMPRISES      5746-CB100
                        RESIDES IN      PRD3.OPTIONAL
                        .
                        .
                        .
                        .
M041I FUNCTION COMPLETED
1S55I LAST RETURN CODE WAS 0000
EOL RETRACE MAX.RETURN CODE=0000
```

```

* $$ JOB JNM=MIGRATE4,CLASS=A,DISP=H,PRI=3,NTFY=YES
// JOB MIGRATE4 ISOLATE PRODUCT IN SUBLIB
// LIBDEF PHASE,SEARCH=(IJSYSRS.SYSLIB,PRD1.BASE)
// EXEC LIBR,PARM='MSHP'
DEFINE S=PRD3.COBOL
CONNECT S=PRD3.OPTIONAL : PRD3.COBOL
MOVE $$$COG44.* REPLACE=YES LIST=YES
MOVE FCOB*.* REPLACE=YES LIST=YES
MOVE ILA*.* REPLACE=YES LIST=YES
MOVE ILB*.* REPLACE=YES LIST=YES
/*
/&
* $$ EOJ

```

```

* $$ JOB JNM=MIGRATE5,CLASS=A,DISP=H,PRI=3,NTFY=YES
// JOB MIGRATE5 CREATE NEW FORMAT TAPE WITH HISTORY
// LIBDEF PHASE,SEARCH=(IJSYSRS.SYSLIB,PRD1.BASE)
// DLBL IJSYSHF,'TEMP.OLD.HIST.FILE'
// EXTENT SYS018,242
// ASSGN SYS018,242
// DLBL PRD3,'TEMP.OLD.LIB'
// EXTENT,SYSWK2,1,0,4000,200000
// PAUSE MOUNT TAPE FOR BACKUP ON 1B1
// ASSGN SYS006,1B1
// MTC REW,SYS006
// EXEC MSHP
RESIDENCE PROD=CB1E44 PROD=PRD3.COBOL
BACKUP PRODUCT=CB1E44 ID='COBOL' PROD
/*
// MTC REW,SYS006
/&
* $$ EOJ

```


CHAPTER 9. MISCELLANEOUS JOBS

OVERVIEW

The job samples listed below are shown on the following pages:

FASTCOPY Contains FCOPYFB jobs which dump and restore entire volumes excluding expired files.

Backup/Restore in the Interactive Interface environment
Demonstrates how to backup and restore the files which are an integral part of the Interactive Interface

Create standalone DUMP program Shows how to create the standalone DUMP program on tape.

FASTCOPY SAMPLES

```
* $$ JOB JNM=FCBKPRES,CLASS=0,DISP=L,PRI=3
// JOB FCBKPRES DUMP SYSRES VOLUME
// UPSI 000
// LIBDEF PHASE,SEARCH=IJSYSRS.SYSLIB,TEMP
// TLBL UOUT,'DOSRES.DUMP',99/365
// ASSGN SYS004,DISK,TEMP,VOL=DOSRES,SHR
// ASSGN SYS005,1B0,D0 OUTPUT TAPE - 6250 BPI
* MOUNT DOSRES TAPE BACKUP ON 1B0
// PAUSE
// EXEC FCOPYFB
DUMP VOLUME NOEXPIRED LIST OV=DOSRES
/*
// MTC RUN,SYS005
/&
* $$ EOJ
```

```
* $$ JOB JNM=FCBKPWK1,CLASS=0,DISP=L,PRI=3
// JOB FCBKPWK1 DUMP SYSWK1 VOLUME
// UPSI 000
// LIBDEF PHASE,SEARCH=IJSYSRS.SYSLIB,TEMP
// TLBL UOUT,'SYSWK1.DUMP',99/365
// ASSGN SYS004,DISK,TEMP,VOL=SYSWK1,SHR
// ASSGN SYS005,1B0,D0 OUTPUT TAPE - 6250 BPI
* MOUNT SYSWK1 BACKUP TAPE ON 1B0
// PAUSE
// EXEC FCOPYFB
DUMP VOLUME NOEXPIRED LIST OV=SYSWK1
/*
// MTC RUN,SYS005
/&
* $$ EOJ
```



```

* $$ JOB JNM=FCRSTRES,CLASS=0,DISP=L,PRI=3
// JOB FCRSTRES RESTORE DOSRES
// UPSI 000
// LIBDEF PHASE,SEARCH=IJSYSRS.SYSLIB,TEMP
// TLBL UIN
// PAUSE
// ASSGN SYS004,1B0,D0      6250 !!!!!
// ASSGN SYS005,DISK,VOL=DOSRES,SHR
* MOUNT DOSRES BACKUP ON 1B0
// PAUSE
// MTC REW,SYS004
// EXEC FCOPYFB
  RESTORE VOLUME LIST OV=DOSRES
/*
// MTC RUN,SYS004
/&
* $$ EOJ

```

```

* $$ JOB JNM=FCRSTWK1,CLASS=0,DISP=L,PRI=3
// JOB FCRSTWK1 RESTORE SYSWK1
// UPSI 000
// LIBDEF PHASE,SEARCH=IJSYSRS.SYSLIB,TEMP
// TLBL UIN
// PAUSE
// ASSGN SYS004,1B0,D0      6250 !!!!!
// ASSGN SYS005,DISK,VOL=SYSWK1,SHR
* MOUNT SYSWK1 BACKUP ON 1B0
// PAUSE
// MTC REW,SYS004
// EXEC FCOPYFB
  RESTORE VOLUME LIST OV=SYSWK1
/*
// MTC RUN,SYS004
/&
* $$ EOJ

```

BACKUP/RESTORE IN THE INTERACTIVE INTERFACE ENVIRONMENT

As explained in Volume 1, the ICCF library (DTSFILE) now has a close relationship with two VSAM files. These VSAM files are the 'VSE.CONTROL.FILE' and the 'VSE.TEXT.REPSTORY.FILE'. Therefore, your normal DTSFILE backup procedures should now include a step to backup these VSAM files.

The following is a sample job to backup both the DTSFILE and the VSAM datasets. Step 1 will backup the DSTFILE and Step 2 will backup the two VSAM datasets.

Note: This job requires **two** tapes.

Backup Interactive Interface Files

```
* $$ JOB JNM=ICCFVSAM,CLASS=0,DISP=D
// JOB ICCFVSAM  BACKUP THE EXISTING ICCF LIBRARY AND VSAM FILES
* STEP 1. ==> BACKUP THE ICCF LIBRARY
// TLBL DTSBKUP,'ICCF BACKUP'
* MOUNT THE ICCF BACKUP TAPE ON THE TAPE DRIVE
* PLEASE '0  // ASSGN SYS005,1Bx,D0'
// PAUSE
// EXEC PROC=DTRICCF
// EXEC DTSUTIL
BACKUP
/*
// MTC RUN,SYS005
* STEP 2. ==> BACKUP THE VSAM DATASETS.
// LIBDEF PHASE,SEARCH=IJSYSRS.SYSLIB
// DLBL IJSYSUC,'VSESP.USER.CATALOG',,VSAM
* PLEASE REMOVE THE ICCF BACKUP TAPE AND
* MOUNT THE VSAM BACKUP TAPE ON THE SAME DRIVE.
// PAUSE  ENTER '0' TO CONTINUE
// TLBL  VSMBKUP,'VSAM.BACKUP.FILE1'
// EXEC IDCAMS,SIZE=AUTO
      BACKUP      ( -
                  VSE.CONTROL.FILE  -
                  VSE.TEXT.REPSTORY.FILE  -
                  ) -
                  BUFFERS(3) -
                  STDLABEL(VSMBKUP)

/*
/&
* $$ EOJ
```

Restore ICCF User Libraries

ICCF libraries used by the system:

- 1, 2 ICCF-provided code
- 3, 4, 5, 6 Public libraries
- 10 System Administrator
- 50 to 69 Interactive Interface-provided code

All other libraries defined are user libraries.

```
* $$ JOB JNM=EX421RIF,CLASS=0,DISP=0
// JOB EX$@|RIF RESTORE SELECTED ICCF LIBRARIES
// TLBL DTSRSTR,'ICCF.BACKUP'
// ASSGN SYS004,1B0                    INPUT TAPE (FROM BACKUP)
// MTC REW,SYS004
// ASSGN SYS010,DISK,VOL=SYSWK1,SHR    DTSFILE EXTENT # 1
// ASSGN SYS011,DISK,VOL=SYSWK2,SHR    DTSFILE EXTENT # 2
// EXEC DTSUTIL
RESTORE TO(21) FROM(21)
RESTORE TO(22) FROM(22)
RESTORE TO(23) FROM(23)
RESTORE TO(36) FROM(24)
RESTORE TO(37) FROM(25)
/*
// MTC RUN,SYS004
/&
* $$ EOJ
```

Restore of Interactive Interface VSAM Files

```
* $$ JOB JNM=RESTIIV,CLASS=0,DISP=D,PRI=3
// JOB RESTIIV RESTORE II VSAM FILES
// LIBDEF PHASE,SEARCH=IJSYSRS.SYSLIB
// ASSGN SYS004,1B0,D0
* MOUNT TAPE ON DEVICE 1B0
* IF NOT POSSIBLE ENTER // ASSGN SYS004,1BX,D0
// PAUSE
// TLBL VSMREST,'VSAM.BACKUP.FILE1'
// EXEC IDCAMS,SIZE=AUTO
  RESTORE OBJECTS ( -
                  * -
                  ) -
                  CATALOG(VSESP.USER.CATALOG) -
                  STDLABEL(VSMBKUP) -
                  USECLASS(0 0) -
                  BUFFERS(3)
/*
/&
* $$ EOJ
```

CREATE STANDALONE DUMP PROGRAM

F6 006 // JOB PAUSEF6
DATE 11/28/84,CLOCK 13/13/40
F6-006 // Pause
6 // EXEC DOSVSDMP
F6-006 4G01D SELECT ONE OF THE FOLLOWING FUNCTIONS:

1 CREATE STANDALONE DUMP PROGRAM
2 PRINT DUMP TAPE
3 PRINT SDAID TAPE
R END DOSVSDMP PROCESSING

6 1
F6-006 4G04D SPECIFY OUTPUT DEVICE IN THE FORM SYSNNN OR CUU
6 1B2
F6 006 4G09I DUMP PROGRAM HAS BEEN CREATED
*F6 006 1I00D READY FOR COMMUNICATION.
*F6-006
6
F6 006 EOJ PAUSEF6
DATE 11/28/84,CLOCK 13/20/06,DURATION 00/06/25

CHAPTER 10. VM/VSE

OVERVIEW

The job samples listed below are shown on the following pages:

VM/PASSTHRU Shows a VM/PASSTHRU session and demonstrates how to switch back and forth between CMS and VSE/SP 2.1 Interactive Interface environments.

CMS EXEC Samples Contains various CMS EXECs.

JSCAN1 creates a tape which you can use as input to the job control scanner.

JSCAN2 shows a simple method to replace a data string (for example, a job control statement or parameter) in a CMS file program.

Router Facility Briefly describes the Router facility and how you can replace it by using VSE/POWER 2.2.0.

VM/PASSTHRU

The following is a sample PASSTHRU EXEC to illustrate the use of the different parameters:

```
&TRACE
&TYPE PASSTHRU
&TYPE PF10 FOR COPY FEATURE
&TYPE PF11 FOR TEMP DISCONNECT
&TYPE PF12 FOR PERM DISCONNECT
EXEC PASSTHRU ISBOE * PVM 10 24 80 PF11 PF12
&EXIT
```

Figure 1. Sample PASSTHRU EXEC

Sample Session Using VM/PASSTHRU

The following pages give an example of using VM/PASSTHRU from a CMS userid. Each framed box is a screen output, and input is highlighted either before or after the box, depending on the execution logic.

The first step is to have access to a **PASSTHRU EXEC**. In the PROFILE EXEC of the CMS user, have the following statement:

CP SET PF11 IMM PASSTHRU

This means as soon as PF11 is entered, the user will execute the PASSTHRU EXEC. Assume the sample PASSTHRU EXEC in Figure 1 on page 337 was used.

==>>
PF11

PASSTHRU
PF10 FOR COPY FEATURE
PF11 FOR TEMP DISCONNECT
PF12 FOR PERM DISCONNECT

VM/370 ONLINE

IIIIIIII	BBBBBBB	MM	MM
IIIIIIII	BBBBBBBB	MMM	MMM
II	BB B	MM	MM MM
II	BBBBBBB	MM	MM
II	BB B	MM	MM
IIIIIIII	BBBBBBBBB	MM	MM
IIIIIIII	BBBBBBB	MM	MM
VV	VV	MM	MM
VV	VV	MMM	MMM
VV	VV	MM	MM MM
VV	VV	MM	MM
VVVV	VM/SP 3.1	MM	MM
VV	4331-02	MM	MM

RUNNING

==>>
ENTER

===>>

D VSESP21

D VSESP21
DIALED TO VSESP21 081

Running

IESADMS01

VSE/SP 2.1 ONLINE

XXX	XXX	XXXXXX	XXXXXXXXXX	+++	XXXXXX	XXXXXX
XXX	XXX	XXXXXXXXXX	XXXXXXXXXX	+++	XXXXXXXXXX	XXXXXXXXXX
XXX	XXX	XXX	XXX	+++	XXX	XXX XX
XXX	XXX	XXXXXXXXXX	XXXXXXXXXX	+++	XXXXXXXXXX	XXXXXXXXXX
XXX	XXX	XXXXXXXXXX	XXXXXXXXXX	+++	XXXXXXXXXX	XXXXXXXXXX
XXXXXX		XXX	XXX	+++	XXX	XXX
XXXX	XXXXXXXXXX	XXXXXXXXXX	+++	XXXXXXXXXX	XXX	
XX	XXXXXX	XXXXXXXXXX	+++	XXXXXX	XXX	

Your terminal is D081 and its name in the network is D081
Today is 01/15/85 To sign on to DBDCCICS -- enter your

USER-ID....==> VENT The name by which the system knows you
PASSWORD....==> YOUR CONFIDENTIAL ACCESS CODE.

PF1=HELP 2=TUTORIAL 4=REMOTE APPLICATIONS
10=NEW PASSWORD

===>>

ENTER Interactive Interface userid (VENT) and password

IESADMSL.IESEADM

VSE/SP 2.1 FUNCTION SELECTION

Enter the number of your selection and press the ENTER key:

- 1 Installation
- 2 Resource Definition
- 3 Operations
- 4 Problem Handling
- 5 Program Development
- 6 Command Mode

PF1=HELP

3=SIGN OFF

6=ESCAPE(U)

9=Escape(m)

==> 6

==>> 6 (to use ICCF)

\$libr

...+...1...+...2...+...3...+...4...+...5...+...6...+...7.

PROGRAM FUNCTION SET

READY

==>>>

\$libr (to execute the librarian interactively)

```
ld lib=prd1
...+....1....+....2....+....3....+....4....+....5....+....6. <==MORE=
L001A ENTER COMMAND OR END
*ENTER DATA?
```

===>>

ld lib=prd1 (librarian command)

```
...+....1....+....2....+....3....+....4....+....5....+....6. <==MORE=
LD LIB=PRD1
STATUS  DISPLAY          LIBRARY=PRD1          DATE: 85-01-22
                                         TIME: 09:08
-----
FILE-ID      : VSE.PR11.LIBRARY
CREATION DATE : 85-01-08
SUBLIBRARIES :      1
LOCATION (BAM) : DEVICE=FBA  VOLID=DOSRES  BLK# = 138818 - 202057
LIBRARY BLOCK : SIZE= 1024 BYTES  DATA SPACE= 988 BYTES
TOTAL  SPACE  :   31620 LIBRARY BLOCKS  (100 %)
USED    SPACE  :   20096 LIBRARY BLOCKS  ( 64 %)
DELAYED SPACE  :         0 LIBRARY BLOCKS  (  0 %)
FREE    SPACE  :   11524 LIBRARY BLOCKS  ( 36 %)
-----
SUBLIBRARY  CREATION  SPACE    NO. OF    USED    DELAYED    % LIBR.
```

===>>

PF10 (for hardcopy to CMS file PASSTHRU DATA)

==>>

PF11 (for returning to CMS without logging off from Interactive Interface

USER REQUESTED TEMP DISCONNECT
R; T=2.68/8.01 09:19:04

==>>

TYPE PASSTHRU DATA A1

...+....1....+....2....+....3....+....4....+....5....+....6. <==MORE=

LD LIB=PRD1

STATUS DISPLAY

LIBRARY=PRD1

DATE: 85-01-22

TIME: 09:08

FILE-ID : VSE.PRD1.LIBRARY
CREATION DATE : 85-01-08
SUBLIBRARIES : 1
LOCATION (BAM) : DEVICE=FBA VOLID=DOSRES BLK# = 138818 - 202057
LIBRARY BLOCK : SIZE= 1024 BYTES DATA SPACE= 988 BYTES
TOTAL SPACE : 31620 LIBRARY BLOCKS (100 %)
USED SPACE : 20096 LIBRARY BLOCKS (64 %)
DELAYED SPACE : 0 LIBRARY BLOCKS (0 %)
FREE SPACE : 11524 LIBRARY BLOCKS (36 %)

SUBLIBRARY	CREATION	SPACE	NO. OF	USED	DELAYED	% LIBR.
------------	----------	-------	--------	------	---------	---------

==>>

PF11 (takes you back to Interactive Interface where you were,
no sign on to Interactive Interface required).

```
PASSTHRU
PF10 FOR COPY FEATURE
PF11 FOR TEMP DISCONNECT
PF12 FOR PERM DISCONNECT
```

...+....1....+....2....+....3....+....4....+....5....+....6. <==MORE=

LD LIB=PRD1

STATUS DISPLAY

LIBRARY=PRD1

DATE: 85-01-22

TIME: 09:08

```
-----
FILE-ID       : VSE.PRD1.LIBRARY
CREATION DATE  : 85-01-08
SUBLIBRARIES   : 1
LOCATION (BAM)  : DEVICE=FBA    VOLID=DOSRES  BLK# = 138818 - 202057
LIBRARY BLOCK  : SIZE= 1024 BYTES  DATA SPACE= 988 BYTES
TOTAL SPACE   : 31620 LIBRARY BLOCKS (100 %)
USED SPACE    : 20096 LIBRARY BLOCKS ( 64 %)
DELAYED SPACE  : 0 LIBRARY BLOCKS ( 0 %)
FREE SPACE    : 11524 LIBRARY BLOCKS ( 36 %)
-----
```

```
-----
SUBLIBRARY  CREATION  SPACE    NO. OF    USED    DELAYED    % LIBR.
-----
```

==>>

PF12 (to take you back to CMS and logoff from Interactive Interface

```
SESSION ENDED BY USER
R; T=0.33/0.80 09:21:02
```

CMS EXEC SAMPLES

Scanner Tape Creation

- Sample EXEC for Scanner Tape Creation.
- Sample EXEC has the following assumptions:
 - Tape is attached as 181.
 - Tape will be written at 6250 BPI.
 - A CMS EXEC has been created.
 - Includes a &1 &2 FN FT FM for each file.
 - Is a list of CMS files to be scanned.

```

&TRACE
*****
*
* JSCAN1 EXEC - MOVES CMS FILES TO TAPE FOR USE WITH VSE JCL SCANNER
*
* EXEC CALLS ITSELF TO DO THE ACTUAL MOVE FROM LIST IN CMS EXEC FILE
*
*****
* &1/2/3 IS FN/FT/FM FOR THE FILE BEING MOVED
* IF 3 PARAMETERS THIS IS NESTED CALL
*****
<u>&IF &INDEX EQ 3 &GOTO -MOVE</u>
<u>-STR</u>
*
* Startup
*
* Set file count to 0
*
<u>EXECIO 1 DISKW JSCAN1 GLOBAL1 A 1 (STRING 0</u>
*
* is the tape drive 181?
*
<u>TAPE REW</u>
*
* if no 181 DMS message of tape not attached will display
*
<u>&IF &RETCODE NE 0 &EXIT 16</u>
<u>TAPE WTM (DEN 6250</u>
<u>TAPE REW</u>
*
* is there a CMS EXEC file?
*
<u>STATE CMS EXEC *</u>
<u>&IF &RETCODE NE 0 &TYPE - CMS EXEC FILE NEEDED</u>
<u>&IF &RETCODE NE 0 &EXIT 16</u>
*
* execute nested JSCAN1 using CMS EXEC
*
<u>EXEC CMS EXEC JSCAN1</u>
<u>&GOTO -END</u>
<u>-MOVE</u>
*
* which file is this?
*
<u>EXECIO 1 DISKR JSCAN1 GLOBAL1 A 1</u>
<u>&READ VARS &X</u>
<u>&X = &X + 1</u>
*
* place a :READ record on the tape
*

```



```

&STACK :READ &1 &2 &3
EXECIO 1 DISKW JSCAN1 GLOBAL2 A &Y F
FI INMOVE DISK JSCAN1 GLOBAL2 A (LRECL 80 BLOCK 80 RECFM F
FI OUTMOVE TAP1 (LRECL 80 BLOCK 80 RECFM F
*
* move the file to tape
*
MOVE
*
* BSF to position over tape mark
*
TAPE BSF
FI INMOVE DISK &1 &2 &3 (LRECL 80 BLOCK 80 RECFM F
FI OUTMOVE TAP1 (LRECL 80 BLOCK 80 RECFM F
MOVE
TAPE BSF
EXECIO 1 DISKW JSCAN1 GLOBAL1 A 1 (STRING &X
&EXIT
-END
ERASE JSCAN1 GLOBAL2
*
* write a few tape marks
*
TAPE WTM 10
TAPE REW
EXECIO 1 DISKR JSCAN1 GLOBAL1 A 1
&READ VARS &X
*
* tell user how many files were moved
* user already knows the names as they were listed when moved
*
&TYPE JSCAN1 COMPLETED - &X FILES PLACED ON TAPE 181
ERASE JSCAN1 GLOBAL1
&EXIT

```

Assist in Migration of JCL

```
&TRACE
*****
*
* JSCAN2 EXEC - Example of a simple EXEC to Edit a File and
*               replace a data string.
* JSCAN2 &1 &2 &3
*****
* &1/2/3 IS FN/FT/FM FOR THE FILE
*****
<u>&TYPE ENTER DATA STRING TO BE REPLACED</u>
<u>&READ STRING &A</u>
<u>&TYPE ENTER DATA STRING TO BE INSERTED</u>
<u>&READ STRING &B</u>
<u>&C = &CONCAT OF CH/ &A / &B /*</u>
<u>&STACK &C</u>
<u>&STACK FILE</u>
<u>HT</u>
<u>EDIT &1 &2 &3</u>
<u>RT</u>
<u>&TYPE FILE &1 &2 &3 HAS BEEN CHANGED</u>
<u>&EXIT</u>
```

ROUTER FACILITY

How it Worked

- USER USED SUBVSE (or equivalent) to submit to VSE machine
- First 3 characters of POWER JOBNAME were CMS USER identifier
- A Printer/Punch setup was made
- for example, 05D/05E were PSTARTed as class R **and** spooled to the virtual machine - ROUTER
- ROUTER on receiving file in its READER
 - Examines Filename of the file from VM SPOOL BLOK
 - Finds CMS USERID match in the SUB\$VSET DTRTABLE

- Transfers file to READER of identified CMS user **or** if not found - sends file to SYSTEM PRT/PUN

ROUTER Replacement

- USE the LDEST, PDEST, DEST parameters in JECL
- This is a user responsibility just as LST CLASS=R was.
- Sample EXEC CNVDEST to alter JCL to add a DEST parameter.

```

&TRACE
*****
* EXEC CNVDEST
* to convert a jobstream for use with POWER/VSE 2.2 VM Writer Task
* FIELD 1 - FN * FIELD 2 - FT * FIELD 3 - FM - Specify all 3 fields *
*****
&IF &INDEX LT 3 &EXIT 16
&TYPE - ENTER LDEST AND PDEST (USE % FOR PRT IF ONLY PUN)
&READ VARS &L &P
*
* &L & &P are destinations for LDEST and PDEST
*
* LOOP through the file looking for * $$JOB or * $$ JOB
*
&J = 1
-READA
EXECIO 1 DISKR &1 &2 &3 &J
&IF &RETCODE NE 0 &GOTO -DONE
&READ STRING &H
&M = &SUBSTR OF &H 1 7
&M1 = &STRING OF * $$JOB
&IF &M EQ &M1 &GOTO -HITJ
&M = &SUBSTR OF &H 1 8
&M1 = &STRING OF * $$ JOB
&IF &M EQ &M1 &GOTO -HITJ
-MOR2
&J = &J + 1
&GOTO -READA
*
* We have located a card image that is the POWER JOB card.
*
-HITJ
&MJ =
*
* Handle LDEST parameter
*
&IF .&L EQ .% &GOTO -HITJ0
&IF .&L EQ . &GOTO -HITJ0
&MJ = &CONCAT OF ,LDEST= (*, &L )
-HITJ0
*
* Handle PDEST parameter
*
&IF .&P EQ . &SKIP 1
&MJ = &CONCAT OF &MJ ,PDEST= (*, &P )
&ML = &LENGTH OF &M
-HITJ1
&ML = &ML + 1
&BL = &SUBSTR OF &H 2 1
&CH = &SUBSTR OF &H &ML 1
&IF &CH EQ &BL &GOTO -HITJ1
-HITJ2
&ML = &ML + 1

```

```

&CH = &SUBSTR OF &H &ML 1
&IF &CH NE &BL &GOTO -HITJ2
-HITJ3
&MK = 80 - &ML
&REST = &SUBSTR OF &H &ML &MK
&ML = &ML - 1
&FIRST = &SUBSTR OF &H 1 &ML
&H = &CONCAT OF &FIRST &MJ &REST
*
* Write the Adjusted Card Image Back
*
EXECIO 1 DISKW &1 &2 &3 &J (STRING &H
&GOTO -MOR2
-DONE
&TYPE FILE ' &1 &2 &3 ' HAS BEEN CONVERTED.
&EXIT

```

CHAPTER 11. DASD SHARING

This chapter does not discuss all aspects of a DASD sharing environment and its related problems, because the scenario has basically not changed since this facility was announced in VSE/Advanced Functions 1.2.0. Therefore, only new functions and considerations are discussed.

A summary of the chapter is given below: **Functional Enhancements** - briefly describes the new parameters of VSE/Advanced Functions 2.1

ASI Procedures - highlights the essential DASD Sharing related parameters in ASI IPL and JCL procedures.

POWER Generation - shows a sample of a POWER generation with the Shared Spooling feature.

System-supplied Procedures - describes procedures which are delivered with VSE/SP 2.1 and how to adapt them to a DASD Sharing environment.

Two Interactive Interface Environments - Procedure Modification - this is an extension of the previous subchapter. It discusses procedural adaptations which are necessary in context with the Interactive Interface (already including some aspects of a second Interactive Interface).

Two Interactive Interfaces - VSAM Files - discusses more implications of a second full-function Interactive Interface with respect to VSAM files.

Librarian in VSAM Managed Space - briefly discusses libraries in VSAM managed Space.

POWER Considerations - describes DASD sharing related to VSE/POWER.

FUNCTIONAL ENHANCEMENTS

- Changes in DASD Sharing
 - The number of sharing systems can now be specified (up to 31).
 - The size of the lock file can be specified.
- Important
 - All sharing systems should specify the same number of CPUs.

- General Rules
 - All other rules of DASD Sharing (for example, usage of LOCKOPTIONS) are unchanged.
- Warning
 - Problems can arise if all sharing systems want to use the Interactive Interface.
 - A significant part of this chapter deals with the points-to-ponder for installing and using a second Interactive Interface.

Basic Functions

New IPL Parameters

Command	Parameter	Remarks
DLF	NCPU=	Number of CPUs in DASD Sharing Environment (2-31)
	NBLK=	Size of FBA Device Lock File
	NCYL=	Size of CKD Device Lock File

- DLF Command

A change of any of the DLF parameters requires an immediate re-IPL of all sharing CPUs.

- NCPU Parameter

- Maximum number of sharing CPUs is 31.

- The default is 4.

- NBLK Parameter

- The default size of the lock file on a FBA device is 80 blocks.

- NCYL Parameter

- The default size of the lock file on a CKD device is 1 cylinder.

Format of DLF Command

Operation	Operands
DLF	{UNIT=cuu VOLID=volid} [,CYL=n ,BLK=n] [,DSF={Y N}] [,NCPU=n] [,NCYL=n ,NBLK=n] [,TYPE={N F}]

ASI PROCEDURES

ASI Procedures CPU 1

IPL ASI Procedure

```
* $$ JOB JNM=IPLVAM,CLASS=A,DISP=D,NTFY=YES,PRI=9
* $$ LST CLASS=R,DEST=(,MERZ),DISP=D
* $$ PUN CLASS=R,DEST=(,MERZ),DISP=D
// JOB IPLVAM CATALOG IPL PROCEDURE FOR ASI-VAM
// EXEC LIBR,PARM='MSHP'
ACCESS SUBLIB=IJSYSRS.SYSLIB
CATALOG $IPLVAM.PROC          DATA=YES REPLACE=YES
01F,$$A$SUP3,VIO=2560K,VPOOL=512K,P,NOLOG,VSIZE=35M
ADD 00C,3505

ADD 240,FBA,SHR
ADD 241,FBA,SHR
ADD 242,FBA,SHR
```

ADD 243,FBA

ADD FFE,PRT1 ICCF DUMMY PRINTER, DO NOT DELETE
DLF VOLID=SYSWK2,BLK=253488,NBLK=512,DSF=N,TYPE=N,NCPU=16
SET ZONE=EAST/01/00

DEF SYSREC=SYSWK2
DPD VOLID=SYSWK2,BLK=360946,NBLK=43910,DSF=N
DPD VOLID=SYSWK1,BLK=377766,NBLK=17050
DPD VOLID=DOSRES,BLK=121954,NBLK=16864
DLA NAME=CPU1LAB,VOLID=DOSRES,BLK=55118,NBLK=744,DSF=N
SYS JA=YES
SVA PSIZE=534K,SDL=250,GETVIS=616K
/+
CONNECT SUBLIB=IJSYSRS.SYSLIB : PRD2.SAVE
COPY \$IPLVAM.PROC REPLACE=YES LIST=YES
/*
// EXEC LISTLOG
/&
* \$\$ EOJ

JCL ASI Procedure

```
* $$ JOB JNM=JCLVAM,CLASS=A,DISP=D,NTFY=YES,PRI=9
* $$ LST CLASS=R,DEST=(,MERZ),DISP=D
* $$ PUN CLASS=R,DEST=(,MERZ),DISP=D
// JOB JCLVAM CATALOG JCL PROCEDURES FOR ASI-VAM
// EXEC LIBR,PARM='MSHP'
ACCESS SUBLIB=IJSYSRS.SYSLIB
CATALOG $OJCLVAM.PROC          DATA=YES REPLACE=YES
ALLOC 1, BG=1536K,F9=2560K,FB=5120K
ALLOC 2, F7=1024K,F2=7168K,FA=1024K
ALLOC 3, F5=4096K,F6=4096K,F8=1024K
ALLOC S, F3=2048K,F4=0256K,F1=1024K
ALLOC R, BG=0128K,F9=0064K,F8=0064K
ALLOC R, FA=0064K,FB=0064K

STDOP JCANCEL=NO
// EXEC PROC=STDLABEL          CALLS STDLABUP AND STDLABUS
// EXEC PROC=SETSDL
START F1
STOP
DVCDN 243
// JOB BGINIT
// EXEC PROC=LIBASGN,PID=BG
// PWR PRELEASE RDR,VTAMCPU1
// EXEC IESWAIT,PARM='02'
// PWR PRELEASE RDR,ICCFCPU1
/+
CATALOG $1JCLVAM.PROC          DATA=YES REPLACE=YES
// JOB PWRSHRD - START POWER FOR SHARED ENVIRONMENT
// EXEC PROC=LIBASGN,PID=F1
// EXEC PWRSHRD
SET SYSID=1
FORMAT=NO
PSTART BG,0A,A
READER=FEC
PRINTERS=FEE,FEF
PUNCHES=FED

/*
/+
CATALOG $2JCLVAM.PROC          DATA=YES REPLACE=YES
// JOB F2INIT
// EXEC PROC=LIBASGN,PID=F2
/+
```

```
CONNECT SUBLIB=IJSYSRS.SYSLIB : PRD2.SAVE
COPY $0JCLVAM.PROC REPLACE=YES LIST=YES
COPY $1JCLVAM.PROC REPLACE=YES LIST=YES
COPY $2JCLVAM.PROC REPLACE=YES LIST=YES
COPY $3JCLVAM.PROC REPLACE=YES LIST=YES
COPY $4JCLVAM.PROC REPLACE=YES LIST=YES
COPY $5JCLVAM.PROC REPLACE=YES LIST=YES
COPY $6JCLVAM.PROC REPLACE=YES LIST=YES
COPY $7JCLVAM.PROC REPLACE=YES LIST=YES
COPY $8JCLVAM.PROC REPLACE=YES LIST=YES
COPY $9JCLVAM.PROC REPLACE=YES LIST=YES
COPY $AJCLVAM.PROC REPLACE=YES LIST=YES
COPY $BJCLVAM.PROC REPLACE=YES LIST=YES
/*
// EXEC LISTLOG
/&
* $$ EOJ
```

ASI Procedures CPU 2

IPL ASI Procedure

```
* $$ JOB JNM=IPLVUK,CLASS=A,DISP=D,NTFY=YES,PRI=9

* $$ LST CLASS=R,DEST=(,MERZ),DISP=D
* $$ PUN CLASS=R,DEST=(,MERZ),DISP=D
// JOB IPLVUK CATALOG IPL PROCEDURE FOR ASI-VUK
// EXEC LIBR,PARM='MSHP'
ACCESS SUBLIB=IJSYSRS.SYSLIB
CATALOG $IPLVUK.PROC          DATA=YES REPLACE=YES
01F,$$A$SUP3,VIO=2560K,VPOOL=512K,P,LOG,VSIZE=32M
ADD 00C,3505

ADD 240,FBA,SHR
ADD 241,FBA,SHR
ADD 242,FBA,SHR
ADD 243,FBA

ADD FFE,PRT1      ICCF DUMMY PRINTER, DO NOT DELETE
DLF VOLID=SYSWK2,BLK=253488,NBLK=512,DSF=N,TYPE=N,NCPU=16
SET ZONE=EAST/01/00
DEF SYSCAT=DOSRES
DEF SYSREC=SYSWK2
DPD VOLID=SYSWK3,BLK=85696,NBLK=38912,DSF=N
DPD VOLID=SYSWK2,BLK=404856,NBLK=38912
DLA NAME=CPU2LAB,VOLID=SYSWK3,BLK=8128,NBLK=768,DSF=N
SYS JA=YES
SVA PSIZE=534K,SDL=250,GETVIS=616K
/+
CONNECT SUBLIB=IJSYSRS.SYSLIB : PRD2.SAVE
COPY $IPLVUK.PROC REPLACE=YES LIST=YES
/*
// EXEC LISTLOG
/&
* $$ EOJ
```

JCL ASI Procedure

* \$\$ JOB JNM=JCLVUK,CLASS=A,DISP=D,NTFY=YES,PRI=9

```
* $$ LST CLASS=R,DEST=(,MERZ),DISP=D
* $$ PUN CLASS=R,DEST=(,MERZ),DISP=D
// JOB JCLVUK CATALOG JCL PROCEDURES FOR ASI-VUK
// EXEC LIBR,PARM='MSHP'
ACCESS SUBLIB=IJSYSRS.SYSLIB
CATALOG $0JCLVUK.PROC DATA=YES REPLACE=YES
ALLOC 1, BG=2048K,F8=1536K,F5=5120K
ALLOC 2, F2=7168K
ALLOC 3, F7=512K,F6=7680K
ALLOC S, F3=2048K,F4=0256K,F1=1024K
ALLOC R, BG=0128K,F8=0064K
ALLOC R, F7=0128K,F6=0256K

STDOPT JCANCEL=NO
// EXEC PROC=STDLAB2 CALLS STDLABP2 AND STDLABS2
// EXEC PROC=SETSDL
START F1
STOP
// JOB BGINIT
// EXEC PROC=LIBASGN,PID=BG
// PWR PRELEASE RDR,VTAMCPU2
// EXEC IESWAIT,PARM='04'
// PWR PRELEASE RDR,ICCFCPU2
/+
CATALOG $1JCLVUK.PROC DATA=YES REPLACE=YES
// JOB PWRSHRD - START POWER FOR SHARED ENVIRONMENT
// EXEC PROC=LIBASGN,PID=F1
// EXEC PWRSHRD
SET SYSID=2
FORMAT=NO
PSTART BG,OA,B
READER=FEC
PRINTERS=FEE
PUNCHES=FED

/*
/+
CATALOG $2JCLVUK.PROC DATA=YES REPLACE=YES
// JOB F2INIT
// EXEC PROC=LIBASGN,PID=F2
/+
```

```

CONNECT SUBLIB=IJSYSRS.SYSLIB : PRD2.SAVE
COPY $0JCLVUK.PROC REPLACE=YES LIST=YES
COPY $1JCLVUK.PROC REPLACE=YES LIST=YES
COPY $2JCLVUK.PROC REPLACE=YES LIST=YES
COPY $3JCLVUK.PROC REPLACE=YES LIST=YES
COPY $4JCLVUK.PROC REPLACE=YES LIST=YES
COPY $5JCLVUK.PROC REPLACE=YES LIST=YES
COPY $6JCLVUK.PROC REPLACE=YES LIST=YES
COPY $7JCLVUK.PROC REPLACE=YES LIST=YES
COPY $8JCLVUK.PROC REPLACE=YES LIST=YES
/*
// EXEC LISTLOG
/&
* $$ EOJ

```

POWER GENERATION

```

* $$ JOB JNM=PWRSHRSP,CLASS=0,DISP=D
* $$ LST CLASS=R,DISP=D,DEST=(*,IANH)
// JOB PWRSHRSP ASSEMBLE NEW POWER FOR SHARED ENVIRONMENT
LIBDEF PHASE,CATALOG=PRD2.CONFIG
// OPTION CATAL,NODECK
// EXEC ASSEMBLY
PWR      TITLE 'VSE/POWER - PWRSHRSP GENERATION '
PWRSHRD  POWER                                     C
                                                C
          ACCOUNT=YES,
          SHARED=(Q,A),    *** SHARED SPOOLING ***      C
          SYSID=1,         *** SHARED SPOOLING ***      C
          TIME=(5,1,60),   *** SHARED SPOOLING ***      C
          TRACKGP=0
          END
/*
// EXEC LNKEDT,PARM='MSHP'
/&
* $$ EOJ

```

Note: You can override SYSID with the SET SYSID statement during POWER autostart.

If only one POWER is generated, it uses the same time slices for **all** sharing systems. To overcome this problem, choose one of two methods:

1. Generate different POWER versions. Specify different **TIME** (and optionally SYSID) parameters for each version.

2. Dynamically change the time slices in storage. Please refer to "Handling POWER Time Slices" on page 409.

SYSTEM-SUPPLIED PROCEDURES

VSE/SP 2.1 provides a set of job skeletons (in ICCF library 59), ICCF members which are included (via /INCLUDE) during ASI processing, and a number of procedures which are called by ASI procedures and partition start-up jobs.

Lets first have a look at those procedures (ICCF members) which may have an influence on a DASD Sharing environment. Here is a short summary. For a graphic overview, see Figure 2 on page 365.

ASIBATCH Establishes LIBDEF chains and makes UR assignments for all partitions except **F1** (POWER) and **BG**.

Called during ASI JCL processing.

DTRSYSWK Assigns the workfiles SYSLNK, SYS001 to SYS004 for all partitions except **F1** (POWER).

Called during ASI JCL processing.

STDLABEL Loads the System Standard Labels. Calls two nested procedures: **STDLABUP** and **STDLABUS**.

Called by ASI JCL procedure \$0JCLxxx.

STDLABUP Loads (adds) VSAM labels, created via the Interactive Interface, into the System Standard Label Area.

Called by procedure **STDLABEL**

STDLABUS Loads user labels into the appropriate PARSTD label area. This procedure contains no labels after initial install of VSE/SP 2.1.

Called by procedure **STDLABEL**.

LIBDEF Establishes the LIBDEF chains for the BG partition.

Called by ASI JCL procedure \$0JCLxxx.

SETSDL Loads additional phases into the SVA, that is, phases which are not automatically loaded during the IPL SVA loadlist processing.

Called by ASI JCL procedure \$0JCLxxx.

DTRPOWR Performs the assignments of the POWER files.

Called by ASI JCL procedure \$1JCLxxx.

DTRICCF Makes the assignments for the ICCF DTSFILE.

Called by procedure \$1JCLxxx and the ICCF start-up job.

Note that this procedure is also called by several Interactive Interface-created jobs which manipulate or access the DTSFILE.

DTRINFOA Performs the assignments for the INFO/ANALYSIS workfiles.

Called by the ICCF start-up job.

DTRCICST Provides the label and assign information for DTSANALS. It also makes the necessary assignments for several CICS files.

Called by the ICCF start-up job.

Structure

On the following two pages, you can find a graphical representations of the most important job skeletons, ICCF members, and procedures provided with VSE/SP 2.1.

In "Clean-up" on page 367, you can find hints how to clean up these procedures.

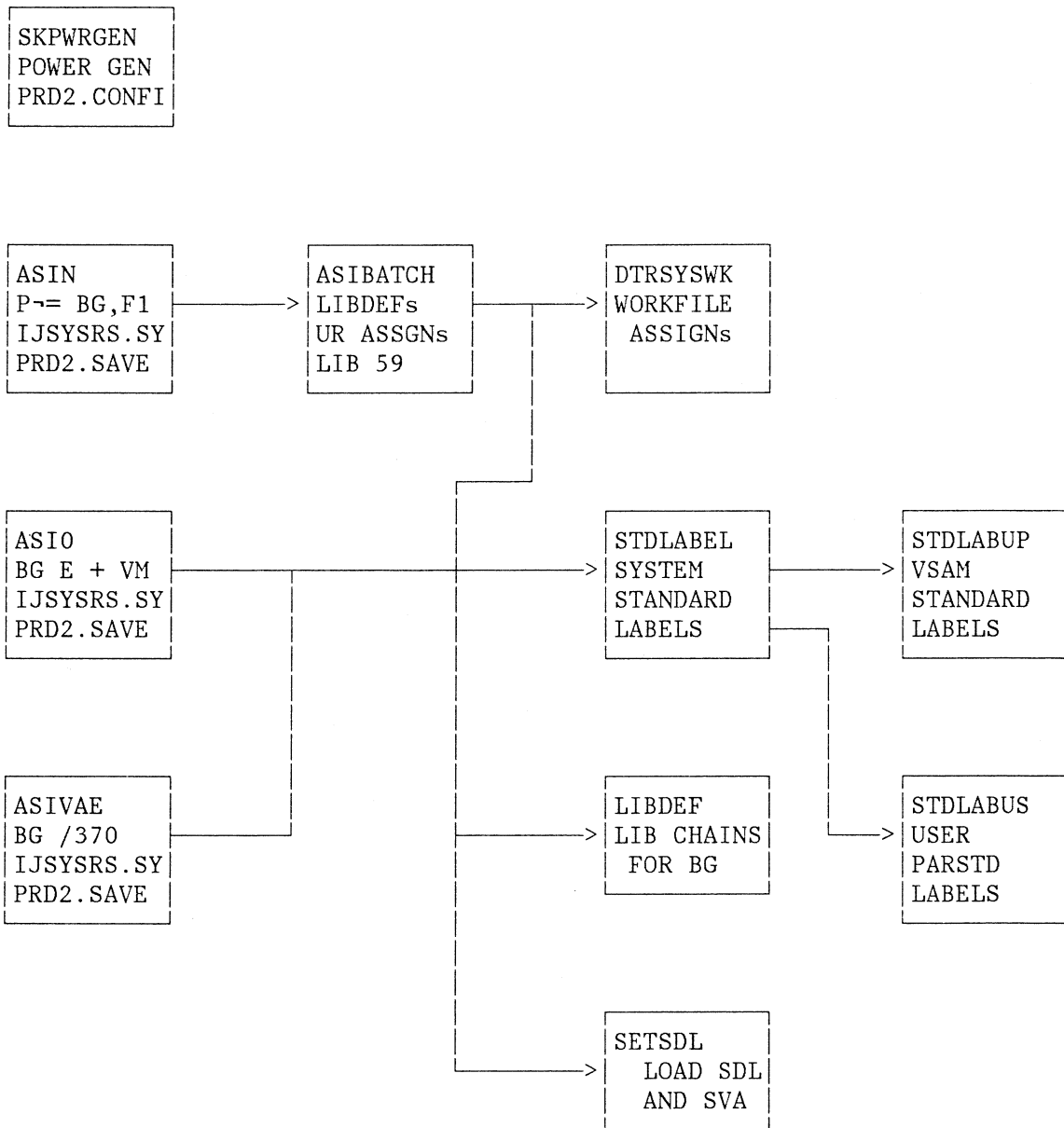


Figure 2 (Part 1 of 2). Overview

SKELETON

ICCF MEMBERS

PROCEDURES

PROCEDURES

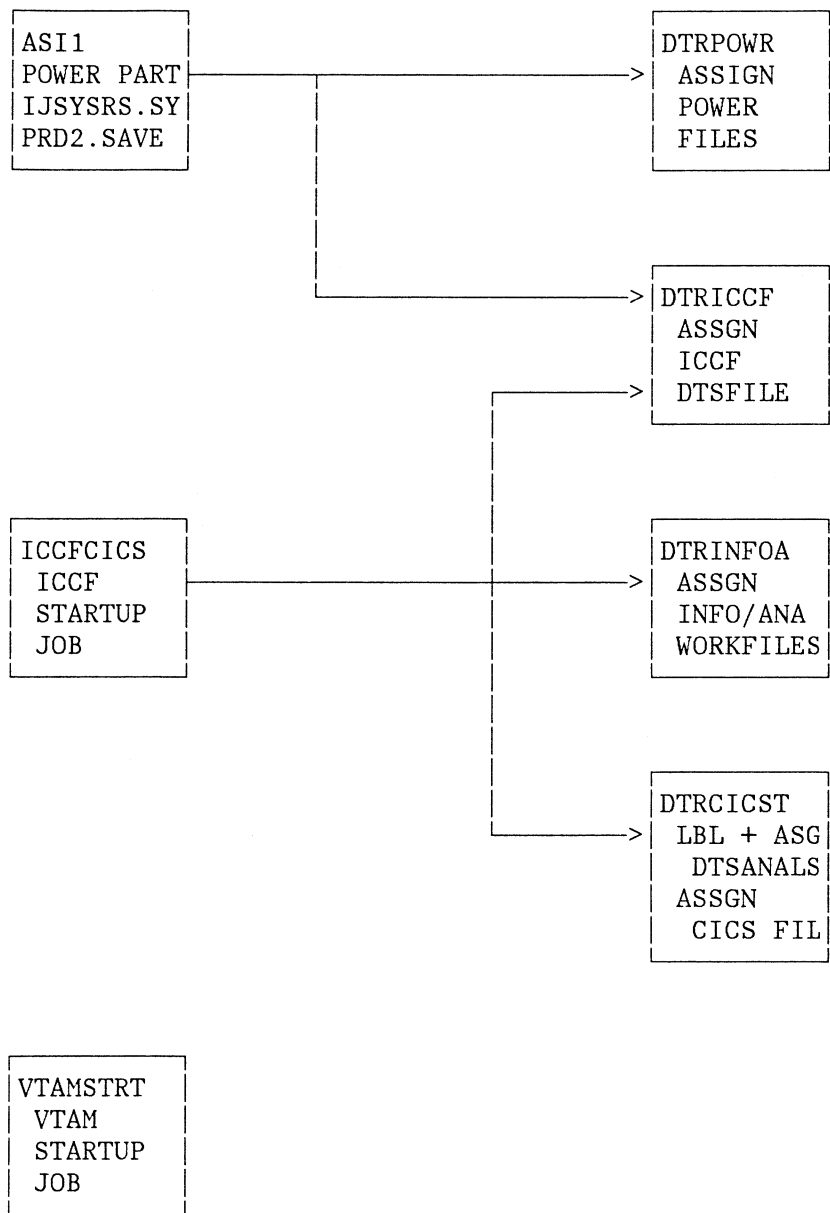


Figure 2 (Part 2 of 2). Overview

Clean-up

Many of the procedures which are shown in the previous section, are set up to cope with different DASD layouts. This is required by the initial install process, and is achieved by the usage of conditional job control. Since conditional JCL and **duplicate** sets of JCL statements mean both, additional overhead and more difficult maintenance, these superfluous statements which do not apply to your environment, should be removed.

Keep in mind that most of these procedures have to be replicated in a DASD sharing environment n times (n depending on the number of sharing systems), which of course implies different contents and member names.

Therefore, if you want to use the procedure scheme as provided by VSE/SP 2.1, a clean-up of these procedures to make adaptation and maintenance easier, seems the right thing to do.

An alternative is to combine all these procedures into one which is **system independent** and thus eases maintenance significantly. It actually makes all procedures which fall into the above described category obsolete. An example of how to code such a procedure is given in "LIBASGN Procedure" on page 378.

There is a second category of procedures which deal with label processing, loading the SVA, and assigning the ICCF DTSFILE. These procedures should be kept and - if necessary - be modified accordingly, because they are an integral part of the system startup (that is, ASI) concept.

It is obvious that these procedures have to be adapted to the according system environment. As far as the loading of labels is concerned they are identical for all sharing systems except for those files which have to be allocated exclusively for each system, for example, RECORDER FILE, HARDCOPY FILE, and so on. Loading of phases into the SVA actually may be the same on each system unless there is a need - mostly triggered by performance considerations - to make different sets of programs sharable within a system.

However, there are two problematic procedures in case you are installing multiple Interactive Interfaces. These are STDLABUP and DTRICCF. See "Prepare Procedures for DASD Sharing" on page 375 and "Two Interactive Interfaces - Procedure Modifications" on page 384 for details. But first a look on how to clean up the respective procedures.

Summary of Cleaned-up Procedures

Procedures should be cleaned-up / replaced

Procedure/ ICCF Mbr.	Action / Comments	Sample provided	Replaceable by LIBASGN
ASIBATCH	Can be tailored to your environment. Establishes identical LIBDEF chains for all partitions except BG and F1.	No	Yes
DTRSYSWK	Can be tailored to your environment. Makes identical work-file assignments for all partitions.	Yes	Yes
LIBDEF	Can be tailored to your environment. Establishes the LIBDEF chains for the BG partition.	No	Yes
DTRPOWR	Has to be modified if the location (size) of the POWER files has been changed.	Yes	Yes
DTRINFOA	No clean-up necessary. Has to be modified only if INFO/ANA files are relocated.	Yes	Yes
DTRCICST	Modification only needed if CICS CSD file is relocated.	Yes	Yes

Procedures Have to Be Maintained

Procedure	Action / Comments	Sample provided	Replaceable by LIBASGN
STDLABEL	Has to be adapted if the basic system layout is modified, that is, files are extended or relocated.	No	No
STDLABUP	Minimum adaption is required for the workfiles. <u>All file-ids have to start with %% instead of %.</u>	Yes	No
STDLABUS	Can be used to load partition dependent labels.	No	No
SETSDL	Can be modified to load additional phases into the SVA.	No	No
DTRICCF	Has to be modified if the location (size) of the ICCF DTSFILE <u>has been changed</u> <u>Note: Maintain this procedure under all circumstances.</u>	Yes	No

Samples of Cleaned-up Procedures

The logic of the conditional job control statements imbedded in the VSE/SP 2.1-provided procedures is always the same:

- Find out the type of DASD VSE/SP 2.1 resides on. This is accomplished by calling the nested procedure DTRIPARM which was created during initial install.
- Use different sets of job control statements depending on the DASD device type.

The procedures shown on the following pages have been cleaned up for an **IBM 3370**. All job control statements which should be kept are underscored.

Please keep in mind that cleaning up the procedures does **not** mean that they are yet adapted to a DASD sharing environment. This will be discussed in "Prepare Procedures for DASD Sharing" on page 375.

MEMBER=DTRCICST.PROC SUBLIB=IJSYSRS.SYSLIB AND
PRD2.SAVE

```
// SETPARM DASD=''
// EXEC PROC=DTRIPARM,DASD
// IF DASD GT 3340 THEN
// GOTO L3350
// DLBL IJSYS02,'%DOS.WORKFILE.SYS002.RECOVER',0,VSAM,
//          CAT=VSESPUC,RECSIZE=4096,
//          DISP=(NEW,DELETE),RECORDS=(30,50)
// EXTENT SYS002,SYSWK1
// DLBL IJSYS01,'%DOS.WORKFILE.SYS001.RECOVER',0,VSAM,
//          CAT=VSESPUC,RECSIZE=4096,
//          DISP=(NEW,DELETE),RECORDS=(30,50)
// EXTENT SYS001,SYSWK2
// ASSGN SYS001,DISK,VOL=SYSWK2,SHR      DTSFILE RECOVERY EXTENT 1
// ASSGN SYS002,DISK,VOL=SYSWK1,SHR      DTSFILE RECOVERY EXTENT 2
// ASSGN SYS018,DISK,VOL=DOSRES,SHR      CICS SD FILES
// GOTO EXIT
/. L3350
// DLBL IJSYS01,'%DOS.WORKFILE.SYS001.RECOVER',0,VSAM,
//          CAT=VSESPUC,RECSIZE=4096,
//          DISP=(NEW,DELETE),RECORDS=(30,50)
// EXTENT SYS001,DOSRES
// DLBL IJSYS02,'%DOS.WORKFILE.SYS002.RECOVER',0,VSAM,
//          CAT=VSESPUC,RECSIZE=4096,
//          DISP=(NEW,DELETE),RECORDS=(30,50)
// EXTENT SYS002,SYSWK1
// ASSGN SYS001,DISK,VOL=DOSRES,SHR      DTSFILE RECV EXT 1
// ASSGN SYS002,DISK,VOL=SYSWK1,SHR      DTSFILE RECV EXT 2
// ASSGN SYS018,DISK,VOL=SYSWK1,SHR      CICS SD FILES
/. EXIT
```

MEMBER=DTRICCF.PROC SUBLIB=IJSYSRS.SYSLIB AND
PRD2.SAVE

```
// SETPARM DASD=''
// EXEC PROC=DTRIPARM,DASD
// IF DASD GT 3340 THEN
// GOTO L3350
// ASSGN SYS010,DISK,VOL=SYSWK3,SHR        DTSFILE EXTENT 1
// GOTO EXIT
/. L3350
// ASSGN SYS010,DISK,VOL=SYSWK1,SHR    DTSFILE EXTENT 1
/. EXIT
```

Note: Keep DTRICCF (and update it accordingly) since it is called implicitly by various Interactive Interface functions.

MEMBER=DTRINFOA.PROC SUBLIB=IJSYSRS.SYSLIB AND
PRD2.SAVE

```
// ASSGN SYS016,DISK,VOL=SYSWK1,SHR    INFO/ANA MGMT FILE
// ASSGN SYS017,DISK,VOL=SYSWK1,SHR    INFO/ANA ROUT FILE
```

MEMBER=DTRPOWER.PROC SUBLIB=IJSYSRS.SYSLIB AND
PRD2.SAVE

```
// SETPARM DASD=''
// EXEC PROC=DTRIPARM,DASD
// IF DASD GT 3340 THEN
// GOTO L3350
// ASSGN SYS000,DISK,VOL=DOSRES,SHR      POWER ACCOUNT FILE
// ASSGN SYS001,DISK,VOL=DOSRES,SHR      POWER QUEUE FILE
// ASSGN SYS002,DISK,VOL=SYSWK1,SHR      POWER DATA FILE 1
// ASSGN SYS003,DISK,VOL=SYSWK4,SHR      POWER DATA FILE 2
// GOTO EXIT
/. L3350
// ASSGN SYS000,DISK,VOL=SYSWK1,SHR      POWER ACCOUNT FILE
// ASSGN SYS001,DISK,VOL=DOSRES,SHR      POWER QUEUE FILE
// ASSGN SYS002,DISK,VOL=SYSWK1,SHR      POWER DATA FILE 1
/. EXIT
```

MEMBER=DTRSYSWK.PROC SUBLIB=IJSYSRS.SYSLIB AND
PRD2.SAVE

```
// SETPARM DASD=''
// EXEC PROC=DTRIPARM,DASD
// IF DASD GT 3340 THEN
// GOTO L3350
ASSGN SYSLNK,DISK,VOL=DOSRES,SHR      SYSTEM LINK FILE
ASSGN SYSO01,DISK,VOL=SYSWK1,SHR      SYSTEM WORK FILE 1
ASSGN SYSO02,DISK,VOL=SYSWK1,SHR      SYSTEM WORK FILE 2
ASSGN SYSO03,DISK,VOL=SYSWK1,SHR      SYSTEM WORK FILE 3
ASSGN SYSO04,DISK,VOL=SYSWK1,SHR      SYSTEM WORK FILE 4
// GOTO EXIT
/. L3350
ASSGN SYSLNK,DISK,VOL=DOSRES,SHR      SYSTEM LINK FILE
ASSGN SYSO01,DISK,VOL=SYSWK1,SHR      SYSTEM WORK FILE 1
ASSGN SYSO02,DISK,VOL=SYSWK1,SHR      SYSTEM WORK FILE 2
ASSGN SYSO03,DISK,VOL=SYSWK1,SHR      SYSTEM WORK FILE 3
ASSGN SYSO04,DISK,VOL=SYSWK1,SHR      SYSTEM WORK FILE 4
/. EXIT
```

MEMBER=STDLABUP.PROC SUBLIB=IJSYSRS.SYSLIB AND
PRD2.SAVE

```
// DLBL IJSYSLN, '%%DOS.WORKFILE.SYSLNK', 0, VSAM,
      CAT=VSESPUC, RECSIZE=322,
      RECORDS=(100, 100)
// DLBL DFHSTM, 'CICS.AUTO.STATS.A', 0, VSAM,
      CAT=VSESPUC,
      DISP=(, DELETE)
.
.
.
// DLBL IJSYS01, '%%DOS.WORKFILE.SYS001', 0, VSAM,
      CAT=VSESPUC, RECSIZE=4096,
      DISP=(NEW, DELETE), RECORDS=(50, 100)
// DLBL IJSYS02, '%%DOS.WORKFILE.SYS002', 0, VSAM,
      CAT=VSESPUC, RECSIZE=2000,
      DISP=(NEW, DELETE), RECORDS=(50, 100)
// DLBL IJSYS03, '%%DOS.WORKFILE.SYS003', 0, VSAM,
      CAT=VSESPUC, RECSIZE=4096,
      DISP=(NEW, DELETE), RECORDS=(50, 100)
// DLBL IJSYS04, '%%DOS.WORKFILE.SYS004', 0, VSAM,
      CAT=VSESPUC, RECSIZE=4096,
      DISP=(NEW, DELETE), RECORDS=(50, 100)
// DLBL SORTWK1, '%%DOS.WORKFILE.SYS001.SORT', 0, VSAM,
      CAT=VSESPUC, RECSIZE=4096,
      DISP=(, DELETE), RECORDS=2000
// DLBL IKSYS11, '%%WORK.FILE.N11', 0, VSAM,
      CAT=VSESPUC, RECSIZE=4096,
      DISP=(NEW, DELETE), RECORDS=(20, 20)
.
.
.
// DLBL IKSYS64, '%%WORK.FILE.N64', 0, VSAM,
      CAT=VSESPUC, RECSIZE=4096,
      DISP=(NEW, DELETE), RECORDS=(20, 20)
// DLBL IESCNTL, 'VSE.CONTROL.FILE', , VSAM,
      CAT=VSESPUC
.
.
.
```

Prepare Procedures for DASD Sharing

DASD sharing basically boils down to **one** decision which has to be made individually for each environment:

- Which files **cannot be shared**. This is independent of the individual environment but dictated by system requirements. Samples are:
 - The Page Data Set (DPD command in IPL)
 - The Label Area (DLA command in IPL)
 - Other system related files such as RECORDER FILE, HARDCOPY FILE, and so on. For a complete list of **non-sharable** files, refer to "Overview System File Sharability" on page 399.
- Which files **can (should)** be shared. These are files, if shared file integrity is guaranteed, such as
 - Libraries
 - POWER Spool Files
 - VSAM Files
- Which files are **technically not sharable** but might be defined only once in the sharing environment
 - A typical example is the **system history file**. It is technically not sharable, however it might be desirable to apply service from any system. This implies that only **one** system history file is maintained, and that usage of MSHP must be strictly controlled to avoid concurrent update of the history file.

This decision, of course, also influences which of the procedures can be used across all sharing systems, and which ones have to be duplicated (catalogued under a different name) in order to cope with the individual requirements of each sharing system. The following table provides a short overview of the actions that have to be considered for each procedure. Of course, the names of the procedures - especially for CPU 2 - can be freely chosen. Procedures for CPU 2 which are mandatory (or may be selected) are indicated by an asterisk.

Procedure CPU 1	Action to be taken	Procedure CPU 2
ASIBATCH	May be the same for both CPUs unless different LIBDEF chains are required.	ASIBATCH ASIBCPU2
DTRSYSWK	May be the same for both CPUs unless different workfile assignments are required.	DTRSYSWK DTRSWK2
LIBDEF	May be the same for both CPUs unless a different LIBDEF setup is required for BG.	LIBDEF LIBDEF2
DTRPOWR	Must be the same for both CPUs if POWER files are shared.	DTRPOWR
DTRINFOA	May be different if the DUMP library is not shared.	DTRINFOA DTRIA2
DTRCICST	May be different if multiple Interact. Interfaces are to be used.	DTRCICST DTRCICS2
STDLABEL	Must be different for both CPUs with respect to labels for non-shared files.	STDLAB2
STDLABUP	May be different if multiple Interact. Interfaces are to be used (see Note 1).	STDLABUP STDLABP2
STDLABUS	Should be different in case it is used at all for loading PARSTD labels.	STDLABS2
SETSDL	May be different if different phases have to be made SVA-resident.	SETSDL SETSDL2
DTRICCF	Must be identical under all circumstances if a second Interact. Interface is installed. Adaptations to the respective system environment have to be accomplished by special Techniques. See Note 2.	DTRICCF

Note 1: Please refer to "VSAM Implications" on page 390.

Note 2: This topic is discussed in more detail on the following pages (see also "LIBASGN Procedure" on page 378).

To create - and especially maintain - a rather large set of procedures might be not only cumbersome but also error prone. To make this task easier, the following alternative should be considered: Combine **all** procedures (except those for label processing and SVA loading) into a single one which is system independent.

This means in essence: all system dependent parameters must be distinguished via conditional job control. Since conditional JCL does not yet have a parameter which allows the interrogation of the CPU-ID of the system it is running on, a small ASSEMBLER program has to be developed to do this.

The LIBASGN procedure which is documented on the following pages uses such a routine and replaces most of the procedures previously described thus making documentation and maintenance easier and more transparent.

Please do not forget: special actions have to be taken for DTRICCF and STDLABUP in case a second Interactive Interface is to be installed. Please refer to the descriptions in "DTRICCF Problem" on page 384 and "VSAM Implications" on page 390.

LIBASGN Procedure

The following pages give an example of a procedure containing both symbolic parameters and conditional JCL. It is cataloged as 'LIBASGN.PROC'.

- Function:
 - simplify ASI procedures by maintaining **all** partition start-up labels, LIBDEFs and assigns in one procedure
 - executed from the '\$xJCLyyy' ASI procedures
 - example set up to handle a DASD sharing environment
- Parameter Requirements:
 - requires one parameter - 'PID=xx' - where 'xx' is the partition id of the invoking partition
- Assumptions:

Partition	CPU 1	CPU 2
	Started as (Application)	Started as (Application)
BG	Batch	Batch
FB	Batch (SQL/DS)	—
FA	Batch	—
F9	Batch	—
F8	Batch	Batch
F7	Batch (DL/I)	Batch
F6	Batch (CICS)	CICS
F5	Batch (CICS)	Batch (SQL/DS)
F4	Batch (OCCF)	Batch (OCCF)
F3	VTAM	VTAM
F2	ICCF-CICS	ICCF-CICS
F1	POWER	POWER


```

* $$ JOB JNM=LIBASGN,CLASS=A,DISP=D,NTFY=YES,PRI=9
* $$ LST CLASS=R,DEST=(,MERZ),DISP=D
* $$ PUN CLASS=R,DEST=(,MERZ),DISP=D
// JOB LIBASGN CATALOG LIBASGN PROCEDURE (USE BY ALL ASI PROCS)
// EXEC LIBR,PARM='MSHP'
ACCESS SUBLIB=IJSYSRS.SYSLIB
CATALOG LIBASGN.PROC          EOD=/+ DATA=YES REPLACE=YES
// PROC PID=
// IF PID ^= ' ' THEN
// GOTO EVALPID
/. MISSPID
* *****
* * PARTITION ID IS MISSING *
* * CORRECT THE PROCEDURE CALL *
* * VALID PARTITION IDS ARE BG, F1 TO FB *
* *****
// GOTO EXIT
/. EVALPID
// EXEC CPUID
// IF $RC>4 OR $RC=0 THEN
// GOTO BADCPU
// IF PID > 'F9' THEN
// GOTO BADPID
// IF PID < 'FA' AND PID ^= 'BG' THEN
// GOTO BADPID
// IF PID > 'FB' AND PID < 'F1' THEN
// GOTO BADPID
// IF PID GE 'F1' AND PID LE 'F3' THEN
// GOTO &PID.PID
// IF PID EQ 'F6' THEN
// GOTO &PID.PID
// GOTO BATCH
/. BADCPU
* ***** *
* THE 'CPUID' PROGRAM DOES NOT RECOGNIZE THIS CPU-ID. *
* THE FOLLOWING PAUSE CARD WILL ENABLE YOU TO ENTER THE *
* APPROPRIATE 'EXEC PROC' OR DLBL/ASSGN STATEMENT(S) *
* FOR THIS PARTITION. *
* ***** *
// PAUSE
// GOTO EXIT
/. BADPID
* *****
* * THE PARTITION ID IS INCORRECT *
* * CORRECT THE PROCEDURE CALL *
* * VALID PARTITION IDS ARE BG, F1 TO FB *
* *****
// GOTO EXIT
/. F6PID
// IF $RC=2 OR $RC=4 THEN
// GOTO BATCH
ASSGN SYSIN,FEC,PERM
ASSGN SYSPCH,FED

```

```

ASSGN SYSLST,FEE
LIBDEF OBJ,SEARCH=(PRD2.CONFIG,PRD1.BASE,PRD2.PROD,PRD2.DBASE,      C
                    PRD2.COMM,PRD2.COMM2),PERM
LIBDEF SOURCE,SEARCH=(PRD2.CONFIG,PRD1.BASE,PRD2.PROD,PRD2.DBASE,   C
                    PRD2.COMM,PRD2.COMM2),PERM
// LIBDEF DUMP,CATALOG=SYSDUMP.F6,PERM
ASSGN SYS018,DISK,VOL=SYSWK2,SHR      CICS SD FILES
ASSGN SYS016,DISK,VOL=SYSWK1,SHR      INFO ANAL MANAGEMENT FILE
ASSGN SYS017,DISK,VOL=SYSWK1,SHR      INFO ANAL ROUTINES FILE
// IF $RC=3 THEN
// GOTO F6LIB2
LIBDEF PHASE,SEARCH=(PRD2.CONFIG,PRD1.BASE,PRD2.PROD,PRD2.DBASE,   C
                    PRD2.COMM,PRD2.COMM2),PERM
// GOTO EXIT
/. F6LIB2
LIBDEF PHASE,SEARCH=(ISCLIB.SYSTEM,PRD2.CONFIG,PRD1.BASE,          C
                    PRD2.PROD,PRD2.DBASE,PRD2.COMM,PRD2.COMM2),PERM
// GOTO EXIT
/. F3PID
ASSGN SYSIN,FEC,PERM
ASSGN SYSPCH,FED
ASSGN SYSLST,FEE
LIBDEF OBJ,SEARCH=(PRD2.CONFIG,PRD1.BASE,PRD2.PROD,PRD2.DBASE,      C
                    PRD2.COMM,PRD2.COMM2),PERM
LIBDEF SOURCE,SEARCH=(PRD2.CONFIG,PRD1.BASE,PRD2.PROD,PRD2.DBASE,   C
                    PRD2.COMM,PRD2.COMM2),PERM
// LIBDEF DUMP,CATALOG=SYSDUMP.F3,PERM
// IF $RC=3 OR $RC=4 THEN
// GOTO F3LIB2
LIBDEF PHASE,SEARCH=(PRD2.CONFIG,PRD1.BASE,PRD2.PROD,PRD2.DBASE,   C
                    PRD2.COMM,PRD2.COMM2),PERM
// GOTO EXIT
/. F3LIB2
LIBDEF PHASE,SEARCH=(ISCLIB.SYSTEM,PRD2.CONFIG,PRD1.BASE,          C
                    PRD2.PROD,PRD2.DBASE,PRD2.COMM,PRD2.COMM2),PERM
// GOTO EXIT
/. F2PID
ASSGN SYSIN,FEC,PERM
ASSGN SYSPCH,FED
ASSGN SYSLST,FEE
LIBDEF OBJ,SEARCH=(PRD2.CONFIG,PRD1.BASE,PRD2.PROD,PRD2.DBASE,      C
                    PRD2.COMM,PRD2.COMM2),PERM
LIBDEF SOURCE,SEARCH=(PRD2.CONFIG,PRD1.BASE,PRD2.PROD,PRD2.DBASE,   C
                    PRD2.COMM,PRD2.COMM2),PERM
// OPTION PARSTD=ADD - LOAD CICS LABELS
// DLBL IJSYS01,'%DOS.WORKFILE.SYS001.RECOVER',O,VSAM,              C
//                                     CAT=VSESPUC,RECSIZE=4096,      C
//                                     DISP=(NEW,DELETE),RECORDS=(30,50)
// EXTENT SYS001,DOSRES
// DLBL IJSYS02,'%DOS.WORKFILE.SYS002.RECOVER',O,VSAM,              C
//                                     CAT=VSESPUC,RECSIZE=4096,      C
//                                     DISP=(NEW,DELETE),RECORDS=(30,50)
// EXTENT SYS002,SYSWK1
// OPTION USRLABEL

```

```

// LIBDEF DUMP,CATALOG=SYSDUMP.F2,PERM
ASSGN SYS016,DISK,VOL=SYSWK1,SHR      INFO ANAL MANAGEMENT FILE
ASSGN SYS017,DISK,VOL=SYSWK1,SHR      INFO ANAL ROUTINES FILE
ASSGN SYS001,DISK,VOL=DOSRES,SHR      DTSFILE RECOVERY EXTENT 1
ASSGN SYS002,DISK,VOL=SYSWK1,SHR      DTSFILE RECOVERY EXTENT 2
// IF $RC=3 OR $RC=4 THEN
// GOTO F2LIB2
LIBDEF PHASE,SEARCH=(PRD2.CONFIG,PRD1.BASE,PRD2.PROD,PRD2.DBASE,
PRD2.COMM,PRD2.COMM2),PERM
// EXEC PROC=DTRICCF      ASSGN DTSFILE EXTENT(S)
ASSGN SYS018,DISK,VOL=SYSWK1,SHR      CICS SD FILES
// GOTO EXIT
/. F2LIB2
LIBDEF PHASE,SEARCH=(ISCLIB.SYSTEM,PRD2.CONFIG,PRD1.BASE,
PRD2.PROD,PRD2.DBASE,PRD2.COMM,PRD2.COMM2),PERM
ASSGN SYS010,DISK,VOL=SYSWK3,SHR      DTSFILE EXTENT 1 **CPU2**
ASSGN SYS018,DISK,VOL=SYSWK3,SHR      CICS SD FILES **CPU2**
// GOTO EXIT
/. F1PID
// LIBDEF PHASE,SEARCH=(PRD2.CONFIG,PRD1.BASE),PERM
// LIBDEF SOURCE,SEARCH=(PRD2.CONFIG,PRD1.BASE),PERM
// LIBDEF DUMP,CATALOG=SYSDUMP.F1,PERM
ASSGN SYS000,DISK,VOL=SYSWK1,SHR      POWER ACCOUNT FILE
ASSGN SYS001,DISK,VOL=DOSRES,SHR      POWER QUEUE FILE
ASSGN SYS002,DISK,VOL=SYSWK1,SHR      POWER DATA FILE 1
ASSGN SYS003,DISK,VOL=SYSWK2,SHR      POWER DATA FILE 2
// EXEC PROC=DTRICCF      ASSGN DTSFILE EXTENT(S)
ASSGN SYS011,DISK,VOL=SYSWK2,SHR      DTSFILE EXTENT 2
// IF $RC=1 OR $RC=2 THEN
// GOTO EXIT
ASSGN SYS010,DISK,VOL=SYSWK3,SHR      DTSFILE EXTENT 1 **CPU2**
ASSGN SYS011,UA
// GOTO EXIT
/. BATCH
ASSGN SYSLST,FEE
ASSGN SYSPCH,FED
ASSGN SYSIN,FEC
ASSGN SYSLNK,DISK,VOL=DOSRES,SHR      SYSTEM LINK FILE
ASSGN SYS001,DISK,VOL=SYSWK1,SHR      SYSTEM WORK FILE 1
ASSGN SYS002,DISK,VOL=SYSWK1,SHR      SYSTEM WORK FILE 2
ASSGN SYS003,DISK,VOL=SYSWK2,SHR      SYSTEM WORK FILE 3
ASSGN SYS004,DISK,VOL=SYSWK2,SHR      SYSTEM WORK FILE 4
LIBDEF OBJ,SEARCH=(PRD2.CONFIG,PRD1.BASE,PRD2.PROD,PRD2.DBASE,
PRD2.COMM,PRD2.COMM2),PERM
LIBDEF SOURCE,SEARCH=(PRD2.CONFIG,PRD1.BASE,PRD2.PROD,PRD2.DBASE,
PRD2.COMM,PRD2.COMM2),PERM
// LIBDEF DUMP,CATALOG=SYSDUMP.&PID,PERM
// IF $RC=3 OR $RC=4 THEN
// GOTO BTLIB2
LIBDEF PHASE,SEARCH=(PRD2.CONFIG,PRD1.BASE,PRD2.PROD,PRD2.DBASE,
PRD2.COMM,PRD2.COMM2),PERM
// GOTO EXIT
/. BTLIB2
LIBDEF PHASE,SEARCH=(ISCLIB.SYSTEM,PRD2.CONFIG,PRD1.BASE,

```

```
PRD2.PROD,PRD2.DBASE,PRD2.COMM,PRD2.COMM2),PERM
/. EXIT
/+
CONNECT SUBLIB=IJSYSRS.SYSLIB : PRD2.SAVE
COPY LIBASGN.PROC REPLACE=YES LIST=YES
/*
// EXEC LISTLOG
/&
* $$ EOJ
```

Invoking the LIBASGN Procedure

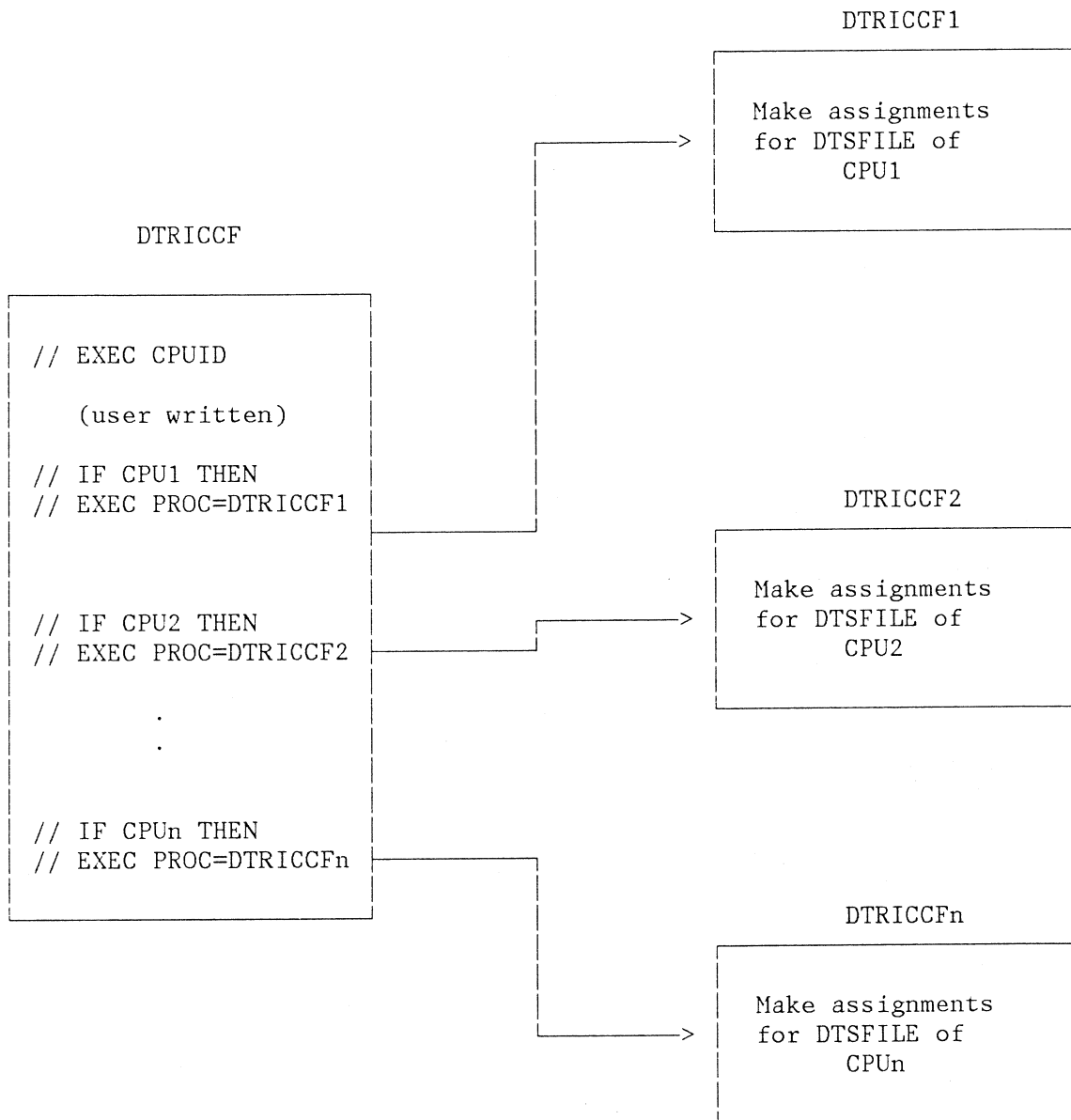
```
CATALOG $0JCLVSL.PROC      DATA=YES REPLACE=YES
ALLOC BG=2048K,F2=5878K
ALLOC F1=756K,F3=2048K
.
.
.
START F1
STOP
// JOB BGINIT
// EXEC PROC=LIBASGN,PID=BG
// PWR PRELEASE RDR,VTAMCPU2
// EXEC IESWAIT,PARM='02'
// PWR PRELEASE RDR,CICSCPU2
/+
CATALOG $1JCLVSL.PROC      DATA=YES REPLACE=YES
// JOB PWRSHRD - START POWER FOR SHARED ENVIRONMENT
// EXEC PROC=LIBASGN,PID=F1
// EXEC PWRSHRD
SET SYSID=2
FORMAT=NO
.
.
.
.
/*
/+
CATALOG $2JCLVSL.PROC      DATA=YES REPLACE=YES
// JOB F2INIT
// EXEC PROC=LIBASGN,PID=F2
/+
CATALOG $3JCLVSL.PROC      DATA=YES REPLACE=YES
// JOB F3INIT
// EXEC PROC=LIBASGN,PID=F3
/+
CATALOG $4JCLVSL.PROC      DATA=YES REPLACE=YES
// JOB F4INIT
// EXEC PROC=LIBASGN,PID=F4
/+
CATALOG $5JCLVSL.PROC      DATA=YES REPLACE=YES
// JOB F5INIT
// EXEC PROC=LIBASGN,PID=F5
/+
```

TWO INTERACTIVE INTERFACES - PROCEDURE MODIFICATIONS

The following description applies only if you want to use more than one full-function Interactive Interface (see also "Two Interactive Interfaces - VSAM Files" on page 398).

DTRICCF Problem

- DTRICCF is a procedure which makes the necessary assignments for the VSE/ICCF DTSFILE. In case of multiple full-function Interactive Interfaces, each Interactive Interface requires a separate DTSFILE.
- It is residing on IJSYSRS.SYSLIB
- Various dialogs of the Interactive Interface imbed this procedure into jobs which are executed immediately, either in an interactive partition of ICCF, or via immediate submission to a batch partition
 - DTRICCF is always picked up from IJSYSRS.SYSLIB, since a LIBDEF PROC SEARCH-chain **cannot** be established during ASI processing.
- Consequence if IJSYSRS is shared
 - DTRICCF has to be modified to allow variable DTSFILE assignments
- Technique used
 - User written program which determines the CPU-ID of the IPLed system
 - Call of nested inner procedures depending on the CPU-ID found
- See the following page for the general structure.



```

* $$ JOB JNM=DTRICCF,CLASS=A,DISP=D,NTFY=YES,PRI=9
* $$ LST CLASS=R,DEST=(,MERZ),DISP=D
* $$ PUN CLASS=R,DEST=(,MERZ),DISP=D
// JOB DTRICCF CATALOG PROCEDURE "DTRICCF"
// EXEC LIBR,PARM='MSHP'
ACCESS SUBLIB=IJSYSRS.SYSLIB
CATALOG DTRICCF.PROC          EOD=/+ DATA=YES REPLACE=YES
// EXEC CPUID
// IF $SRC>4 OR $SRC=0 THEN
// GOTO BADCPU
// IF $SRC=1 OR $SRC=2 THEN
// EXEC PROC=DTRICCF1
// IF $SRC=3 OR $SRC=4 THEN
// EXEC PROC=DTRICCF2
// GOTO DONE
/. BADCPU
* *****
* THE 'CPUID' PROGRAM DOES NOT RECOGNIZE THIS CPU-ID.      *
* THE FOLLOWING PAUSE CARD WILL ENABLE YOU TO ENTER THE    *
* APPROPRIATE 'EXEC PROC' OR ASSGN STATEMENT(S)           *
* FOR THE ICCF LIBRARY.                                     *
* *****
// PAUSE
/. DONE
/+
CONNECT SUBLIB=IJSYSRS.SYSLIB : PRD2.SAVE
COPY DTRICCF.PROC REPLACE=YES LIST=YES
/*
// EXEC LISTLOG
/&
* $$ EOJ

```



```

* $$ JOB JNM=DTRICCF1,CLASS=A,DISP=D,NTFY=YES,PRI=9
* $$ LST CLASS=R,DEST=(,MERZ),DISP=D
* $$ PUN CLASS=R,DEST=(,MERZ),DISP=D
// JOB DTRICCF1 CATALOG PROCEDURE "DTRICCF1"
// EXEC LIBR,PARM='MSHP'
ACCESS SUBLIB=IJSYSRS.SYSLIB
CATALOG DTRICCF1.PROC      EOD=/+ DATA=YES REPLACE=YES
// ASSGN SYS010,DISK,VOL=SYSWK1,SHR      DTSFILE EXTENT 1
// ASSGN SYS011,DISK,VOL=SYSWK2,SHR      DTSFILE EXTENT 2
/+
CONNECT SUBLIB=IJSYSRS.SYSLIB : PRD2.SAVE
COPY DTRICCF1.PROC  REPLACE=YES LIST=YES
/*
// EXEC LISTLOG
/&
* $$ EOJ

```

```

* $$ JOB JNM=DTRICCF2,CLASS=A,DISP=D,NTFY=YES,PRI=9
* $$ LST CLASS=R,DEST=(,MERZ),DISP=D
* $$ PUN CLASS=R,DEST=(,MERZ),DISP=D
// JOB DTRICCF2 CATALOG PROCEDURE "DTRICCF2"
// EXEC LIBR,PARM='MSHP'
ACCESS SUBLIB=IJSYSRS.SYSLIB
CATALOG DTRICCF2.PROC      EOD=/+ DATA=YES REPLACE=YES
// ASSGN SYS010,DISK,VOL=SYSWK3,SHR      DTSFILE EXTENT 1
/+
CONNECT SUBLIB=IJSYSRS.SYSLIB : PRD2.SAVE
COPY DTRICCF2.PROC  REPLACE=YES LIST=YES
/*
// EXEC LISTLOG
/&
* $$ EOJ

```

```

* $$ JOB JNM=CPUID,CLASS=A,DISP=D,NTFY=YES,PRI=9
* $$ LST CLASS=R,DEST=(,MERZ),DISP=D
* $$ PUN CLASS=R,DEST=(,MERZ),DISP=D
// JOB CPUID PHASE DETERMINES CORRECT CPUID DURING ASI PROCESSING
// LIBDEF PHASE,CATALOG=IJSYSRS.SYSLIB
// OPTION CATAL,NODECK,XREF
  PHASE CPUID,*
// EXEC ASSEMBLY,SIZE=128K
CPUID      CSECT
           BALR  5,0
           USING *,5
           LA     8,10
           EXTRACT ID=CPUID,AREA=CPUSTOR,LEN=(8)
CPU1370    EQU   *                SET RETCODE TO 15 IN CASE OF ERROR
           CLC    CPU1A,CPUSTOR    4361-5 RUNNING UNDER VM MODE=370
           BNE    CPU1VM           COMPARE AGAINST VSESP22 CPU-ID
           LA     15,1             TRY VSESP21
           EOJ     RC=(15)         SET RETURN CODE FOR VSESP22
CPU1VM     EQU   *                THAT'S IT FOLKS!!
           CLC    CPU1B,CPUSTOR    4361-5 RUNNING UNDER VM MODE=VM
           BNE    CPU1EMOD         COMPARE AGAINST VSESP21 CPU-ID
           LA     15,2             TRY NATIVE VSE
FOUND1     B      FINISH          SET RETURN CODE FOR VSESP21
CPU1EMOD   EQU   *                4361-5 RUNNING NATIVE VSE
           CLC    CPU1C,CPUSTOR    COMPARE AGAINST REAL CPU-ID
           BNE    CPU2370         TRY VSEUK
           B      FOUND1          SET RETURN CODE FOR NATIVE VSE
CPU2370    EQU   *                4331-2 RUNNING UNDER VM MODE=370
           CLC    CPU2A,CPUSTOR    COMPARE AGAINST VSESPUK CPU-ID
           BNE    CPU2VM          TRY VSESPIH
           LA     15,3             SET RETURN CODE FOR VSESPUK
           B      FINISH
CPU2VM     EQU   *                4331-2 RUNNING UNDER VM MODE=VM
           CLC    CPU2B,CPUSTOR    COMPARE AGAINST VSESPIH CPU-ID
           BNE    CPU2EMOD         TRY NATIVE VSE
FOUND2     LA     15,4             SET RETURN CODE FOR VSESPIH
           B      FINISH
CPU2EMOD   EQU   *                4331-2 RUNNING NATIVE VSE
           CLC    CPU2C,CPUSTOR    COMPARE AGAINST REAL CPU-ID
           BNE    FINISH          UNKNOWN CPU-ID
           B      FOUND2          SET RETURN CODE FOR NATIVE VSE
           DS     OD
CPUSTOR    DS     CL10' '
CPU1A     DC     X'FF0200224361'   VSESP22      MODE=370   RC=1
CPU1B     DC     X'FF0200064361'   VSESP21      MODE=VM    RC=2
CPU1C     DC     X'050224084361'   VSE NATIVE   MODE=E     RC=2
CPU2A     DC     X'FF0200444331'   VSESPUK      MODE=370   RC=3
CPU2B     DC     X'FF0224084331'   VSESPIH      MODE=VM    RC=4
CPU2C     DC     X'020200064331'   VSE NATIVE   MODE=E     RC=4
           END
/*
// EXEC LNKEDT

```

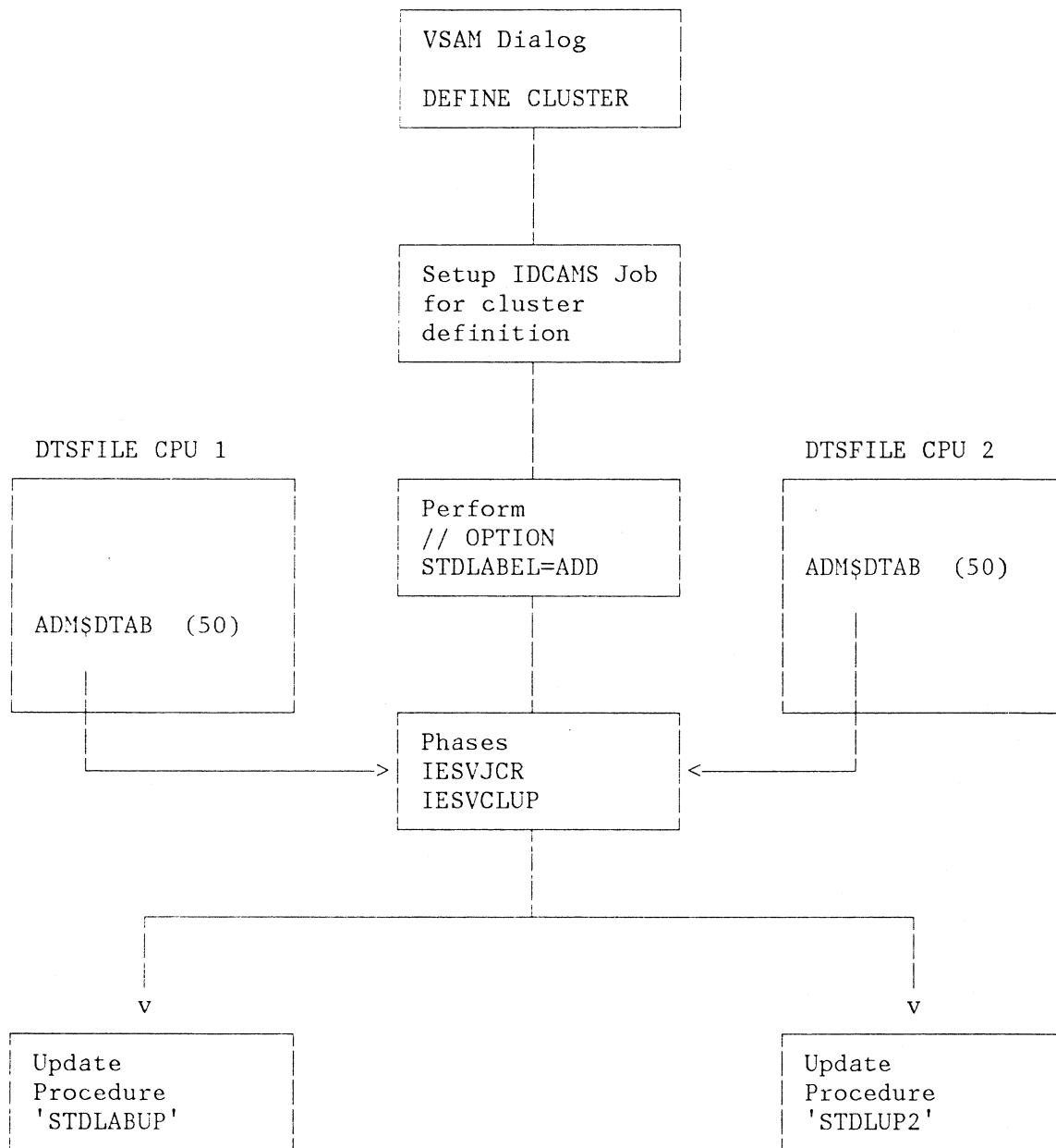
```
/*  
// EXEC LIBR,PARM='MSHP'  
CONNECT SUBLIB=IJSYSRS.SYSLIB : PRD2.SAVE  
COPY CPUID.PHASE REPLACE=YES LIST=YES  
/*  
// EXEC LISTLOG  
/&  
* $$ EOJ
```

VSAM Implications

Description

- The VSAM dialogs for defining clusters perform the following functions:
 - Create an IDCAMS job to define the specified cluster.
 - Create the appropriate DLBL and EXTENT statements, and add them to the active label area.
 - Update the procedure STDLABUP (which is the default procedure delivered with VSE/SP 2.1) with the according label information.
- Scenario
 - In a DASD sharing environment different label procedures are required.
 - If defining a cluster for CPU 2, the following would happen:
 - The cluster would be correctly defined in the specified VSAM catalog.
 - The appropriate label information would be stored in procedure STDLABUP (belonging to CPU 1).
 - Result: the cluster would not be accessible.
- Solution
 - The phases which update the VSAM label procedure pick up the name of this procedure from ICCF member ADM\$DTAB, which is stored in ICCF library 50.
 - This member has to be modified (at least in the DTSFILE of your second CPU), to reflect the procedure name which is used in your BG ASI-procedure for CPU 2.
 - Please refer to "Logic Flow" on page 391 and "Tailor ADM\$DTAB" on page 392 for more details.

Logic Flow



IESADMSL.IESEADM

VSE/SP FUNCTION SELECTION

Enter the number of your selection and press the ENTER key:

- 1 Installation
- 2 Resource Definition
- 3 Operations
- 4 Problem Handling
- 5 Program Development
- 6 Command Mode

PF1=HELP

3=SIGN OFF

4=RETURN

5=REBUILD

6=ESCAPE(U)

9=Escape(m)

==> 51

Path:

IESLIBI

PROGRAM DEVELOPMENT LIBRARY

ENTER the numbers of the Libraries you want to access

PRIMARY.....	50	You may read and write in this library
SECONDARY...	——	You may only read from this library

SPECIFY optionally

PREFIX.....	adm\$d	To list only members whose names begin with the specified prefix
-------------	--------	---

SELECT which library you want to display

OPTION.....	1	1 = Primary Library
		2 = Secondary Library

PF1=HELP 2=REDISPLAY 3=END 4=RETURN 6=ADD MEMBER

IESLIBP

PRIMARY LIBRARY

Page 1 of 1

PRIMARY (READ/WRITE): 50

SECONDARY:

PREFIX: ADM\$D

OPTIONS: 1 = EDIT 2 = CHANGE 3 = PRINT 4 = COPY 5 = DELETE
6 = RENAME 7 = SUBMIT 8 = COMPILE

OPT	MEMBER NAME	NEW NAME	NEW LIB	LAST CHANGED	OWNER	PASSW
1	ADM\$DTAB	_____	_____	11/22/84	POST	_____
-		_____	_____			_____
-		_____	_____			_____
-		_____	_____			_____
-		_____	_____			_____
-		_____	_____			_____

PF1=HELP 2=LIST QUEUE 3=END 4=RETURN 5=REFRESH
PF6=ADD MEMBER 7=BACKWARD 8=FORWARD 9=SORT.DATE 10=SORT.NAME

LOCATE MEMBER ==> _____

*MSG=PF 10/11 SET FOR SHIFT LEFT/RIGHT 4.... MEM=ADM\$DTAB

\$\$\$036**NAME	ID1	ID2	ID3	ID4	ID5	DASD	DATE
CODE SEC	SYS	SER	TYP	SEQ	BSIZE	ESTART	NO
SPLIT BUFSP	CAT	CSIZE	NUMREC	GNAME	IDSIZE	LIB	DISP1
DISP2 PRINO	SECNO	RSize	LIBRY	OVERL	VSYS	LIBUSED	PRICL
SECCL							

\$\$\$036**K	K	K	K	K	K	X	X
X	X	X	K	X	K	X	X
X	X	X	X	X	K	X	X
X	X	X	X	X	X	X	X
X							

\$\$\$036**DASDFREESPACE DOSRES 292826 SYSWK1 487790

LABELPROCEDURE: STDLUP2

\$\$\$036**IJSYSRS	VSE	SYSRES	LIBRARY		3370	99/365
SD	SYSRES	DOSRES	1	0	2	55116
			IJSYSRS	31	STD	
			IJS	YES	YES	

\$\$\$036**IJSYSDL	DOS	LABEL	FILE	CPUID	AREA1	3370	99/365
SD	SYSDLA	DOSRES	1	0		55118	744

Clean-up ICCF Startup Job

Note: This startup procedure was cleaned up for usage of ACF/VTAM. All required statements are underscored.

```
* $$ JOB JNM=CICSICCF,DISP=L,CLASS=2,PRI=3
* $$ LST CLASS=A,DISP=D
// JOB CICSICCF CICS/ICCF STARTUP
// UPSI 00001000 CONSOLE RESPONSES FROM SYSIPT
// LIBDEF *,SEARCH (PRD2.CONFIG,PRD1.BASE,
// PRD2.PROD,PRD2.DBASE)
// LIBDEF DUMP,CATALOG=SYSDUMP.F2
// SETPARM TPMODE=''
// EXEC PROC=DTRIPARM,TPMODE SET TPMODE
// EXEC PROC=DTRCICST ASSGNS FOR CICS FILES
// EXEC PROC=DTRINFOA ASSGNS FOR INFO ANAL FILES
// EXEC PROC=DTRICCF ASSGN FOR DTSFILE
// ASSGN SYS005,UA
// ASSGN SYS006,UA
// ASSGN SYS007,UA
// ASSGN SYS008,UA
// ASSGN SYS009,SYSLOG
// ASSGN SYS013,UA
// ASSGN SYS015,FEF ASSGN PRT FOR IWS FUNCTION
// EXEC DTSANALS RECOVER IF DTSFILE DESTROYED
RECOVER OPT
/*
// EXEC IDCAMS,SIZE=AUTO
// VERIFY FILE(DFHNTRA)
// VERIFY FILE(DFHTEMP)
// VERIFY FILE(DFHSTM)
// VERIFY FILE(DFHSTN)
// VERIFY FILE(IESCNTL)
// VERIFY FILE(IESTRFL)
// VERIFY FILE(IESMSGs)
// VERIFY FILE(IESPRB)
/*
// IF TPMODE=B THEN
// GOTO BTAM
// EXEC IESWAITT WAIT FOR VTAM ACB TO OPEN
// GOTO START
/. BTAM
// ASSGN SYS023,IGN
// ASSGN SYS024,IGN
// ASSGN SYS025,IGN
// ASSGN SYS026,IGN
// ASSGN SYS027,IGN
// EXEC PROC=DTRIBTAM ASSGN BTAM TERMINALS
* AT THIS POINT, YOU MAY OPTIONALLY ASSIGN SYS027 TO
```

C

```
* A 3270 PRINTER SUPPORTED AS A 3286. ENTER:
* "// ASSGN SYS027, CUU" WHERE CUU IS THE PRINTER ADDRESS
// PAUSE
/. START
// EXEC DTSINIT, PARM='SIT=SP, $END'
/*
/&
* $$ EOJ
```

TWO INTERACTIVE INTERFACES - VSAM FILES

This subchapter deals with the VSAM files which are required - besides an additional ICCF DTSFILE - to install a second Interactive Interface. The topics discussed in the following sections are:

Overview System File Sharability - lists the system files which can be shared.

Sharable Interactive Interface VSAM Files - provides the same overview for the files which have to be allocated for the Interactive Interface.

Installation of Second Interactive Interface - deals with the actual installation of a second Interactive Interface.

Overview System File Sharability

Sharable Files		Non-sharable Files
VSE.SYSRES.LIBRARY		CICS.AUTO.STATS.A
VSE.PR1.LIBRARY		CICS.AUTO.STATS.B
VSE.PR2.LIBRARY		CICS.AUXTRACE
VSE.DUMP.LIBRARY		CICS.CSD
VSE.SYSTEM.HISTORY.FILE	3	CICS.DUMPA
VSE.POWER.QUEUE.FILE		CICS.DUMPB
VSE.POWER.DATA.FILE		CICS.MSGUSR
VSE.POWER.ACCOUNT.FILE		CICS.TD.INTRA
VSE.MESSAGES.ONLINE		CU370X.LOAD.FILE
VSAM.MASTER.CATALOG		CU370X.DIAG.FILE
VSESP.USER.CATALOG		DFHTEMP
VSESP.JOB.MANAGER.FILE	3	DOS.LABEL.FILE.CPUID.AREA1
INFO.ANALYSIS.DUMP.MGNT.FILE		ICCF.LIBRARY
INFO.ANALYSIS.EXT.RTNS.FILE		PAGING.DATA.SET.ONE
%DOS.WORKFILE.SYSLNK	1	PAGING.DATA.SET.TWO
%DOS.WORKFILE.SYS001		SYS.NEW.RES
%DOS.WORKFILE.SYS002		VSE.CONTROL.FILE
%DOS.WORKFILE.SYS003		VSE.HARDCOPY.FILE
%DOS.WORKFILE.SYS004		VSE.ONLINE.PROB.DET.FILE
%DOS.WORKFILE.SYS001.RECOVER		VSE.RECORDER.FILE
%DOS.WORKFILE.SYS002.RECOVER		VSE.TEXT.REPSTORY.FILE
%DOS.WORKFILE.SYS001.SORT		VTAM.TRACE.FILE
%DOS.WORK.FILE.N*1	2	WORK.HIST.FILE
%DOS.WORK.FILE.N*2		
%DOS.WORK.FILE.N*3		
%DOS.WORK.FILE.N*4		

Note 1: Filenames have to be preceded by '%%'.

Note 2: * = 1, 2, 3, 4, 5, 6

Note 3: Files should not be used concurrently.

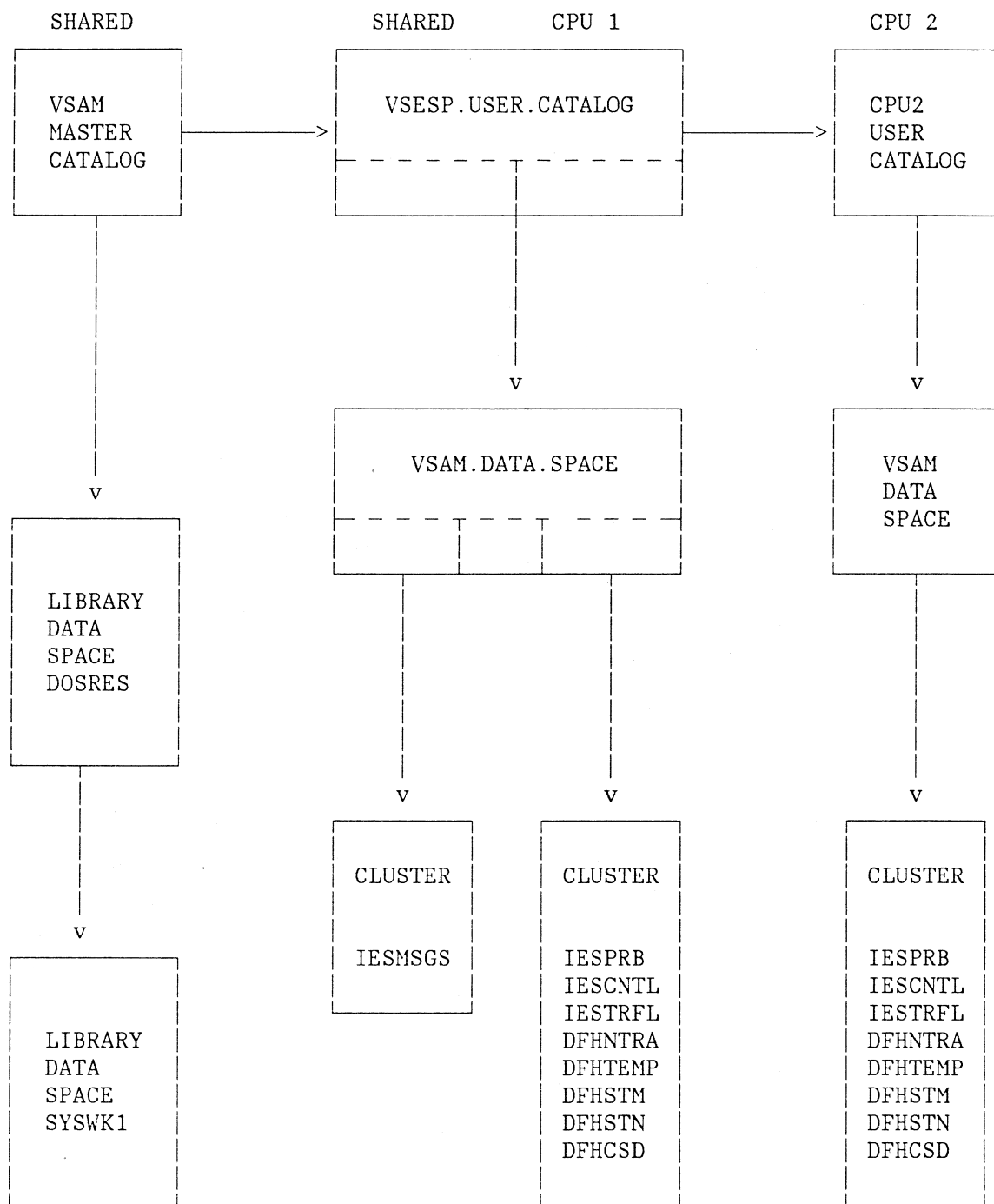
Sharable Interactive Interface VSAM Files

Sharable Files	Non-sharable Files
VSE.MESSAGES.ONLINE	CPU2.USER.CATALOG VSE.CONTROL.FILE VSE.TEXT.REPSTORY.FILE VSE.ONLINE.PROB.DET.FILE CICS.TD.INTRA CICS.AUTO.STATS.A CICS.AUTO.STATS.B CICS.CSD

Installation of Second Interactive Interface

- Establish first Interactive Interface
 - Define ICCF Libraries
 - Establish User Profiles
 - .
 - .
- Create files and clusters for non-sharable data sets
 - A separate user catalog should be used (not VSESPUC). See "VSAM Structure" on page 402.
 - See sample in "VSAM Definitions" on page 403.
- Backup/Restore DTSFILE
 - Backup DTSFILE from CPU 1
 - Restore into DTSFILE of CPU 2
- Copy non-sharable VSAM files
 - See sample in "Loading of VSAM Files" on page 406.

VSAM Structure



VSAM Definitions

```
* $$ JOB JNM=VSAMDEFS,CLASS=0,DISP=D
// JOB VSAMDEFS  DEFINE USER CATALOG, SPACE AND CLUSTERS
* FOR SECOND INTERACTIVE INTERFACE
// EXEC IDCAMS,SIZE=AUTO

/*                                                    */
/*  DEFINE USER CATALOG AND CLUSTERS                */
/*                                                    */
DEFINE USERCATALOG(NAME(CPU2.USER CATALOG)           -
  VOL(SYSWK3)                                         -
  NOTRECOVERABLE                                     -
  ORIGIN(124620)                                     -
  BLK(2976))                                         -
DEFINE SPACE(VOLUMES(SYSWK3)                         -
  ORIGIN(127596)                                     -
  BLK(100000))                                       -
  CATALOG(CPU2.USER.CATALOG)                         -
/*
// EXEC IDCAMS,SIZE=AUTO
/*                                                    */
/*  DEFINE CLUSTER  IESPRB                          */
/*                                                    */
DEFINE CLUSTER(NAME(VSE.ONLINE.PROB.DET.FILE)        -
  FILE(IESPRB)                                       -
  VOL(SYSWK3)                                       -
  RECORDS (300 100)                                 -
  RECORDSIZE (4000 4089)                           -
  INDEXED                                           -
  KEYS(2 0)                                         -
  SHR(2))                                           -
  DATA (NAME (VSE.ONLINE.PROB.DET.FILE.DATA) CISCZ(4096)) -
  INDEX (NAME (VSE.ONLINE.PROB.DET.FILE.INDEX) CISCZ(512)) -
  CATALOG(CPU2.USER.CATALOG)                         -
/*                                                    */
/*  DEFINE CLUSTER  IESCNTRL                        */
/*                                                    */
DEFINE CLUSTER(NAME(VSE.CONTROL.FILE)                -
  FILE(IESCNTRL)                                    -
  RECORDS (120 20)                                  -
  TO (99366)                                         -
  INDEXED                                           -
  KEYS(12 0)                                        -
  RECORDSIZE(100 1000)                             -
  SHR(2)                                             -
  VOLUMES(SYSWK3))                                 -
  DATA (NAME (VSE.CONTROL.FILE.DATA) CISCZ(4096)) -
  INDEX (NAME (VSE.CONTROL.FILE.INDEX))             -
  CATALOG(CPU2.USER.CATALOG)                         -
/*                                                    */
/*  DEFINE CLUSTER  IESTRFL                        */
/*                                                    */
DEFINE CLUSTER(NAME(VSE.TEXT.REPSTORY.FILE)          -
  FILE(IESTRFL)                                     -
```

```

RECORDS (4500 200)
TO (99366)
INDEXED
KEYS(12 0)
RECORDSIZE(500 4080)
SHR(2)
VOLUMES(SYSWK3))
DATA (NAME (VSE.TEXT.REPSTORY.FILE.DATA) CISZ(4096))
INDEX (NAME (VSE.TEXT.REPSTORY.FILE.INDEX))
CATALOG(CPU2.USER.CATALOG)
/*
/* DEFINE CLUSTER DFHNTRA
/*
/*
DEFINE CLUSTER(NAME(CICS.TD.INTRA)
VOL(SYSWK3)
RECORDS (100)
RECORDSIZE (4089 4089)
CISZ (4096)
NONINDEXED
SHR(2))
DATA(NAME(CICS.TD.INTRA.ESDS))
CATALOG(CPU2.USER.CATALOG)
/*
/* DEFINE CLUSTER DFHTEMP
/*
/*
DEFINE CLUSTER(NAME(DFHTEMP)
VOL(SYSWK3)
RECORDS (100)
RECORDSIZE (4089 4089)
CISZ (4096)
NONINDEXED
SHR(2))
DATA(NAME(DFHTEMP.ESDS))
CATALOG(CPU2.USER.CATALOG)
/*
/* DEFINE CLUSTER DFHSTM
/*
/*
DEFINE CLUSTER (NAME(CICS.AUTO.STATS.A)
NONINDEXED
RECORDFORMAT(VARUNB)
RECORDSIZE(133,133)
FOR(0)
REUSE
RECORDS(20,10)
VOLUME(SYSWK3))
DATA(NAME(CICS.AUTO.STATS.A.ESDS))
CATALOG(CPU2.USER.CATALOG)
/*
/* DEFINE CLUSTER DFHSTN
/*
/*
DEFINE CLUSTER (NAME(CICS.AUTO.STATS.B)
NONINDEXED
RECORDFORMAT(VARUNB)
RECORDSIZE(133,133)

```

```

FOR(0)
REUSE
RECORDS(20,10)
VOLUME(SYSWK3))
DATA(NAME(CICS.AUTO.STATS.B.ESDS))
CATALOG(CPU2.USER.CATALOG)
/*
/*  DEFINE CLUSTER  DFHCSD
/*
/*
DEFINE CLUSTER(NAME(CICS.CSD)
INDEXED
RECORDSIZE (100 200)
RECORDS (4000 1000)
KEYS (22 0)
FREESPACE (0 0)
SHR(2)
VOL(SYSWK3))
DATA(NAME(CICS.CSD.DATA))
INDEX (NAME (CICS.CSD.INDEX))
CATALOG(CPU2.USER.CATALOG)
/*
/*  END OF DEFINES
/*
/*
IF MAXCC GT 8 THEN DO
CANCEL JOB
END
ELSE
/*
/. EXIT          END OF VSAM DEFINES
/*
/&
* $$ EOJ

```

Loading of VSAM Files

```
* $$ JOB JNM=VSAMBKUP,CLASS=0,DISP=D,PRI=3
// JOB VSAMBKUP  BACKUP II VSAM FILES
// DLBL IJSYSUC,'VSESP.USER.CATALOG',,VSAM
// TLBL VSMBKUP,'BACKUP.II.FILES'
* // ASSGN SYS005,1BX,D0
// PAUSE
// EXEC IDCAMS,SIZE=AUTO
  BACKUP          (
                    VSE.CONTROL.FILE
                    VSE.TEXT.REPSTORY.FILE
                    CICS.CSD
                    )
                    BUFFERS(3)
                    STDLABEL(VSMBKUP)
/*
// ASSGN SYS005,UA
* READY TAPE DRIVE AGAIN
* // ASSGN SYS004,1BX,D0
// PAUSE
// DLBL IJSYSUC,'CPU2.USER.CATALOG',,VSAM
// TLBL VSMREST,'BACKUP.II.FILES'
// EXEC IDCAMS,SIZE=AUTO
  RESTORE OBJECTS (
                    *
                    )
                    CATALOG(CPU2.USER.CATALOG)
                    STDLABEL(VSMREST)
                    USECLASS(0 0)
                    VOLUMES(111111)
                    BUFFERS(3)
/*
/&
* $$ EOJ
```

LIBRARIES IN VSAM MANAGED SPACE

- Secondary allocation for libraries in VSAM managed space, controlled by a shared catalog, does not work.
- Only way out (very clumsy) in case of a "LIBRARY FULL" condition
 - Shut down all systems which have access to this library.
 - Re-IPL one system with DASDs being defined as non-shared.
 - Catalog several members in order to trigger secondary allocation.
 - Make sure that secondary allocation took place (via LISTCAT).
 - Re-IPL all systems with their original ASI-Procedures.

POWER CONSIDERATIONS

Setup of POWER Classes

- Categorize the POWER input classes. Criteria could be, for instance, priority and the level of DASD sharing (see sample on next page).
- Category 1
 - The CLASS parameter on the POWER JOB card determines the CPU and the partition for which a job is scheduled.
- Category 2
 - To ensure that a job is executed on a certain CPU and in a distinct partition (assuming identical input classes on all systems), the SYSID parameter in the POWER JOB card is used.
- Category 3
 - This category permits limited DASD sharing.
 - Limited DASD sharing means that a job can be executed on any CPU, but only in a certain partition.
- Category 4
 - This category allows full DASD sharing.
 - Jobs can be executed on any CPU and in any partition, which was PSTARTed for this input class.

CPU 1

High Priority —————> Low Priority

Part. ID	Input Classes				Output Class	Usage
	Excl. by Class	Excl. by SYSID	Ltd. Shr.	Full Shr.		
BG	-	-	0	A	A	Batch
FB	B	-	-	A	A	Batch
FA	R	-	-	A	A	Batch
F9	9	-	-	A	A	Batch
F8	H	-	-	A	A	Batch
F7	7	-	-	-	A	DL/I and Batch
F6	6	-	-	-	A	CICS Developm.
F5	-	5	-	-	A	SQL/DS
F4	N/A	N/A	N/A	N/A	N/A	OCCF/NCCF
F3	-	3	-	-	A	VTAM
F2	-	2	-	-	A	CICS / ICCF

CPU 2

BG	-	-	0	A	B	Batch
--	-	-	-	-	-	---
--	-	-	-	-	-	---
--	-	-	-	-	-	---
--	-	-	-	-	-	---
--	-	-	-	-	-	---
--	-	-	-	-	-	---
F5	V	-	-	A	B	Batch
F4	N/A	N/A	N/A	N/A	N/A	OCCF/NCCF
F3	-	3	-	-	B	VTAM
F2	-	2	-	-	B	CICS / ICCF

Handling POWER Time Slices

- The values for the POWER time slices (t1 through t4) in core can be located by adding X'2900' to the POWER partition start address.
- Each time slice occupies a halfword.
- The individual values may be changed via the system console (see sample below).

```

AR 015      1  F9 V    6   768K   768K 3E0000      64K  80000 NO NAME
AR 015      1  FA V    6   512K   512K 560000      64K  A0000 NO NAME
AR 015      1  FB V    6   512K   512K 660000      64K  B0000 NO NAME
AR 015      1  UNUSED      2304K      760000
AR 015      2  F7 V    6   832K   192K  60000      128K  C0000 NO NAME
AR 015      2  F2 V    3  6656K   512K 160000      256K 140000 ICCFCICS
AR 015      2  UNUSED      1280K      860000
AR 015      3  F5 V    6  3264K   832K  60000      128K 120000 NO NAME
AR 015      3  F6 V    4  3072K  1024K 460000      256K  E0000 NO NAME
AR 015      3  UNUSED      1280K      860000
AR 015      S  F1 V    1   640K   384K 9A0000      128K 1C2000 PWRSHRD
AR 015      S  F4 V    5   128K   128K AA0000      64K 180000 NO NAME
AR 015      S  F3 V    2  1280K   768K AE0000      200K 190000 VTAMSTRT
AR 015      S  SVA      1568K  1632K CE0000      162K
AR 015      AVAIL      2304K      2116K
AR 015      TOTAL      32768K      4096K
AR 015 1I40I  READY

```

DSPLY 1,9A2900

AR 015 00050000 003C00B4 00000000 00000000

*AR+015 1I40I READY

ALTER 1,9A2900

15 00050001

AR 015 1I40I READY

DSPLY 1,9A2900

AR 015 00050001 003C00B4 00000000 00000000

AR 015 1I40I READY

APPENDIX A. TABLE OF CONTENTS OF VOLUME 1

<u>Part 1. Overview</u>	1
<u>Chapter 1. Hardware / Software</u>	3
VSE/SP 2.1 Overview	3
What VSE/SP 2.1 is	3
Selected VSE/SP 2.1 Items	3
The Content of the Unique Usability Functions of VSE/SP 2.1 . .	4
Remote Communications	5
Comparison to VSE SIPO/E 1.4.0	5
Comparison to VSE SIPO/E 1.3.1	6
Changed Hardware Support	6
VSE/SP 1.1.0	6
VSE/SP 2.1	7
DASD Layout Comparison	7
Unused Space on CKD Devices	8
IBM 3330	8
IBM 3330	9
IBM 3330	10
IBM 3340	11
IBM 3340	12
IBM 3340	13
IBM 3350	14
IBM 3350	15
IBM 3375	16
IBM 3375	17
IBM 3380	18
Unused Space on FBA Devices	19
IBM 3310.	19
IBM 3310.	20
IBM 3310.	21
IBM 3370-1	22
IBM 3370-1	23
IBM 3370-2	24
Software Component Comparison	24
VSE/SP versus VSE SIPO/E Base	25
Optional Products	26
<u>Part 2. Migrating to VSE/SP 2.1</u>	29
<u>Chapter 2. Designing the System</u>	31
Storage Allocation	31
General Considerations	31
Virtual Storage Layout in E Mode or VM Mode	31
Virtual Storage Layout in 370 Mode	33
Shared/Private Storage Areas	33
GETVIS Area Considerations	37

System GETVIS Area	37
Partition GETVIS Area	38
GETVIS Area Subpooling	39
Allocation of the ICCF Interactive Partitions	39
Allocating Storage for Running in Real Mode	40
SVA Considerations	40
DASD Layout Considerations	41
Step 1 - VSE/SP 2.1 Library Design	41
Step 2 - Page Data Set	44
Step 3 - POWER Files Layout	45
Step 4 - ICCF DTSFILE	46
Step 5 - DUMP Library	47
Step 6 - Lock File	47
Step 7 - Work Files	47
Step 8 - Other System Files	48
Step 9 - Optional Products Files	49
Step 10 - Database and VSAM Files	49
Step 11 - Other Considerations	49
VSE/SP 2.1 Installation Tasks	50
Mandatory Installation Tasks	50
Optional Installation Tasks	50
Installation of Optional Products	50
Hardware Additions after Installation	51
 <u>Chapter 3. What to Do Before You Install</u>	53
Summary of Tasks	53
Individual Tasks	54
Job Control Scanner	54
Incompatible Statements	54
Conditional Job Control	56
VSE/POWER Considerations	56
Library and Sublibrary Definition Jobstreams	57
Scan of Backup Tape	57
Library Migration Table Preparation	57
Backup Jobs for the Current System	57
RESTORE Jobs for VSE/SP 2.1	58
BACKUP Job for VSE/SP 2.1	59
Actual Backup of Libraries	59
Offload of VSE/POWER RDR Queue	59
ASI Procedure Preparation	59
Supervisor Generation Macros	59
ASI Master Procedure	61
IPL Command Changes	61
JCL Procedures	62
Define Standard and User Labels	62
 <u>Chapter 4. What to Do After You Have Installed VSE/SP 2.1</u>	63
Summary of Tasks	63
Individual Tasks	64
Reload the VSE/POWER RDR Queue	64
VSE/ICCF Restore	64
ICCF User Profile Migration	64
ICCF User Profile Migration Utility	65
Program Description	65

Control Statement	65
Adding User Profiles	68
Reserved ICCF Libraries	71
Operation	71
Examples	71
Messages and Codes	73
Scan Backup Tapes	73
Define New Library/Sublibraries	73
Install Library Migration Table	74
Restore Private Libraries	74
Migrating CICS/DOS/VS	74
Multiple CICS/DOS/VS	75
Production CICS (CICS2)	75
ACF/VTAM Considerations	76
Adding Interactive Interface Functions to CICS2	77
Development CICS (CICS1)	77
Single CICS/DOS/VS	77
Backup New Libraries	78
Reinstall Non-VSE/SP 2.1 Products	78
Update ASIPROCS if Needed	78
Set up Fallback Jobstreams	78
General Purpose Jobstreams	79
 <u>Chapter 5. Transferring System Files</u>	 81
POWER RDR Queue Entries	81
History File	83
 <u>Chapter 6. Transferring User Files</u>	 85
Introduction	85
SAM, DAM, ISAM Files	85
No Data Movement Required	85
Data Movement Required	86
Target Volume of Same Type as Source	86
Target Volume Type Different from Source	86
Summary	89
VSAM Files and Data Bases	90
No Data Movement Required	90
Data Movement Required	91
Native VSAM Files	91
DL/I Data Bases	93
SQL/DS Data Bases	94
 <u>Chapter 7. Modifying User Programs</u>	 97
Introduction	97
Job Control Exit (\$JOBEXIT)	97
VSE/POWER Reader Exit	97
Job Accounting Considerations	98
Changes in VSE/POWER Job Accounting Records	98
Job Accounting Partition Table	99
Cross Partition Support	99
Abnormal Termination (STXIT AB)	99
CICS Program Error Program (DFHPEP)	100
Changes in Logical IOCS	100
Sequential Disk Files	100

Device Independent Full Track Support	100
End-of-Extent Exit	101
Tape Files	101
Direct Access Files	102
Printer Files	102
 Chapter 8. <u>Modifying ICCF/CMS Procedures</u>	103
ICCF Procedures and Macros	103
IBM-Supplied Procedures and Macros	103
Procedure Processor Variables	104
Procedure Processor Orders and Macro Orders	104
User-written Procedures/Macros	104
Usage of SLI Statement	105
Generation of SLI Statement	106
SLI Usage	106
CMS EXECs	107
 Chapter 9. <u>Transferring Operations</u>	109
Native VSE Operations	109
ICCF/CICS vs. ICCF/TTF	109
Initialization Procedure	109
Console Communication	110
System Termination Request	110
VSE/ICCF New Operator Commands	111
VM/VSE	111
 Chapter 10. <u>Documenting and Saving the System</u>	113
Documentation	113
System Resources	113
Library and SVA Contents	113
RETRACE System History File	114
VSE/VSAM Catalog Contents	114
VTOC	114
System Startup and Label Procedures	114
BACKUP/RESTORE	115
IPL-able Copy of System	116
Standalone Backup / Restore	116
Tape for Standalone Dumps	116
Operator Instructions	117
Console Functions	117
Interactive Interface	117
Asynchronous Communication	118
New/Changed Commands and Their Output	118
Changes of Utilities	118
Dump Concept	119
System Save Concept	119
Final System Test	120
 Part 3. <u>Migrating to VSE/SP 2.1 under VM/SP</u>	121
 Chapter 11. <u>VM/VSE Interfaces</u>	123
Installing VSE/SP 2.1 under VM/SP	123
VM Directory Entry	123
Additional Options	126

IPL/JCL Procedures	127
Automatic IPL	129
Supervisors	129
IBM 3203-5 Considerations	131
Virtual I/O under VM/SP	131
Use of IPTE	132
VM/VSE Interface Routines	132
Virtual Machine Communication Facility - VMCF	132
Pre-VSE/SP 2.1 SUBVSE and ROUTER	136
VSE/SP 2.1 SUBVSE and VM Writer Task	137
Operator Communication Control Facility - OCCF	138
Interactive User Communication Facility - IUCV	139
Installing VM/VSE Interface Routines on CMS	139
Service for Interface Routines	140
VM/SP CMS Requirements for Interface Routines	140
VSE/SP 2.1 Requirements for Interface Routines	140
POWER VM Writer Tasks	141
Use of VM/PASSTHRU	143
What is VM/PASSTHRU	143
Why VM/PASSTHRU	143
How to Use VM/PASSTHRU	144
Sample Session Without Using VM/PASSTHRU	145
Sample Session Using VM/PASSTHRU	145
 Chapter 12. What is New in VM/VSE	 147
Introduction	147
ICCF PAGEX Setting	147
Disconnected Console	148
Before VSE/SP 2.1	149
With VSE/SP 2.1	149
 Chapter 13. CMS/DOS	 151
Introduction	151
Librarian Restrictions	152
Using DOS Librarian Functions in CMS/DOS	152
DOSLKED Command	153
FCOBOL EXEC	153
VMFDOS Command	154
MACLIB Command	155
FETCH Command	155
LISTIO Command	155
LISTDS Command	155
SET DOS ON (Mode) Command	156
ASSGN Command	156
DLBL Command	156
DOSLIB Command	157
Other CMS/DOS Commands	157
Using CMS/DOS in an VSE/SP 2.1 Environment	157
Alternatives to Lost CMS/DOS Functions	158
Alternative # 1 (DOSLIBs, MACLIBs)	158
Alternative # 2 (VM Server Machine)	160
VSE Sort/Merge	162
Ditto for VSE and VM	162
CMSVSAM	163

<u>Chapter 14. VM/VSE with VAE</u>	165
VAE Considerations	165
General Information	165
Handshaking Subset	165
V=R or V=V	166
Multiple Systems vs. One System with VAE	166
Multiple Systems	166
One System with VAE	167
4K Paging	167
<u>Chapter 15. Other VM/VSE Considerations</u>	169
CMS Users working with the Interactive Interface	169
SLI in ICCF	170
Job Control Scanner	170
Procedures	171
<u>Appendix A. Job Control Scanner</u>	173
Introduction	173
Input Files	173
Program Control	173
DOS/VSE Libraries	174
ICCF Libraries	175
CMS Files	175
Unstructured Job Control	176
Output	176
Printed	176
Punched (to Tape)	177
Storage Reuirements	177
Restrictions	177
<u>Appendix B. Table of Contents of Volume 2</u>	179
<u>Bibliography</u>	183
Publications for VSE/SP 2.1 Component Program Products	183
Selected Publications for VSE/SP 2.1 Optional Program Products	185
<u>Index</u>	187

INDEX

A

ASI master procedure 80
 ASSEMBLY 142, 144
 ASSGN, emulation of
 SYSCLB 200
 SYSRLB 201
 SYSSLB 202

B

BACKUP 308
 BACKUP ALL (VSAM) 308
 BACKUP CL 136
 BACKUP PL 136
 BACKUP RL 136
 BACKUP SL 136
 NOSA,NOSORT,NODSF 136
 SA,NOSORT 135
 backup interactive interface
 files 333
 BACKUP/RESTORE DTSFILE 76
 BACKUP, emulation of 213
 backward migration 194
 BSC
 sample VTAM cluster 321

C

changeover task
 plan
 VTAM books 317
 CMS EXEC samples 345
 assist in migration of JCL 348
 scanner tape creation 345
 communications adaptor
 See ICA
 conditional job control 81
 RC assembler subroutines 81
 sample COBOL program 84

 sample PL/I program 83
 COPYSERV 125, 126, 127, 128, 129,
 130, 131, 132, 133, 134
 COPYSERV, emulation of 216
 CORGZ
 ALLOC 106
 MERGE NRS,PRV
 COPYP NEW 114
 MERGE NRS,RES
 COPYC phasename 110
 MERGE PRV,NRS
 COPYC ALL 115
 MERGE PRV,PRV
 COPYC NEW 116
 COPYR ALL 120
 COPYR modulename 117
 COPYR NEW 119
 COPYR xxx.ALL 118
 COPYS ALL 124
 COPYS NEW 123
 COPYS sublib.ALL 122
 COPYS sublib.bookname 121
 MERGE PRV,RES
 COPYP procedurename 111
 MERGE RES,NRS
 COPY ALL 109
 COPYP ALL 112
 MERGE RES,PRV
 COPYC xxxx.ALL 113
 NEWVOL 108
 CORGZ, emulation of
 ALLOC 204
 COPY ALL 208
 COPYC 248
 COPYI 207
 COPYP 250
 COPYR 252
 COPYS 254
 MERGE 247
 MERGE-COPY Overview 245
 NEWVOL 205
 create standalone DUMP program 336
 cross-domain
 VTAM books 322
 CSERV
 DSPCH phasename 175
 DSPLY ALL 173
 PUNCH xxxx.ALL 174

CSERV, emulation of
 DSPCH 260
 DSPLY 256, 262
 PUNCH 258

D

DASD sharing
 ASI procedures 356
 basic functions 355
 clean-up of supplied
 procedures 367
 DLF command 356
 functional enhancements 353
 general file sharability 399
 highlights 353
 libraries in VSAM managed
 space 406
 new IPL parameters 355
 POWER considerations 407
 POWER generation 362
 POWER input/output classes 407
 POWER time slices 409
 prepare for DASD sharing 375
 structure of supplied
 procedures 364
 supplied procedures 363
 two Interactive Interfaces
 DTRICCF problem 384
 ICCF startup job 396
 installation 401
 procedure modifications 384
 VSAM file definitions 403
 VSAM file loading 406
 VSAM file sharability 400
 VSAM file structure 402
 VSAM files 398
 VSAM implications 390
 DASD sharing highlights 353
 DEFINE VSAM objects
 See VSAM
 DSERV 164
 DSPLY CD 168
 DSPLY CD(XXXX) 169
 DSPLY SDL 166
 DSPLY TD 167
 DSPLYS ALL 165
 DSPLYS PD 172
 DSPLYS RD 170

DSPLYS SD 171
 DSERV, emulation of
 CD (phasename ..) 209

E

ESERV
 DSPCH E.bookname 186
 DSPLY E.bookname 188
 PUNCH E.bookname 187
 ESERV, emulation of 221
 EXEC emulation of
 DSERV 238
 EXEC, emulation of
 CORGZ 237
 CSERV 239
 MAINT 243
 PSERV 240
 RSERV 241
 SSERV 242

F

FASTCOPY samples 330
 files tailoring 24
 forward migration 189

G

generation
 POWER modifications 23
 VTAM books 317

H

HLL and return codes 83
 PL/I and COBOL sample 83

I

ICA
 VTAM books 321
 ICCF 21, 72
 BACKUP/RESTORE DTSFILE 76
 generation changes 21
 SUBMIT procedure 72
 ICCF DTSFILE
 tailoring 29
 IESWAITT
 VTAM application 319
 integrated communications adaptor
 See ICA
 interactive interface 3
 application development 12
 keyword summary 13
 panel path summary 12
 backup of related files 333
 system administration 3
 keyword summary 6
 panel path summary 3
 system operation 15
 keyword summary 16
 panel path summary 15

J

job control scanner 60
 CMS input 65
 create tape 65
 sample output 67
 scan tape 66
 ICCF input 60
 create tape 60
 sample output 61
 scan tape 60
 PL/SSL input 62
 create tape 62
 restore to ICCF 63
 sample output 64
 scan tape 63
 unstructured input 68
 sample output 69
 scan tape 68

L

label procedure skeleton 77
 LIBDEF, emulation of
 FROM 225
 NEW 226
 SEARCH 228
 TO 227
 LIBDROP, emulation of
 ALL 233
 FROM 229
 NEW 231
 SEARCH 232
 TO 230
 LIBLIST, emulation of
 * 235
 ALL 236
 BG|Fn 234
 LIBR
 ACCESS
 CATALOG bookname.sourcetype 146
 CATALOG modulename.OBJ 145
 CATALOG.... 147
 DELETE *.OBJ 158
 DELETE *.PHASE 155
 DELETE *.PROC 163
 DELETE *.sourcetype 160, 161
 DELETE bookname.sourcetype 159
 DELETE modulename.OBJ 156
 DELETE phasename.PHASE 153
 DELETE procedurename 162
 DELETE xxx*.OBJ 157
 DELETE xxxx*.PHASE 154
 LIST *.OBJ 176
 LIST *.PHASE 173
 LIST *.PROC 183
 LIST *.sourcetype 179
 LIST bookname.sourcetype 180,
 182
 LIST modulename.OBJ 178
 LIST phasename.PHASE 175
 LIST procedurename.PROC 185
 LISTD \$\$*.PHASE 167
 LISTD *.OBJ 170
 LISTD *.PHASE 168
 LISTD *.PROC 172
 LISTD *.sourcetype 171
 LISTD XXXX*.PHASE 169
 PUNCH *.sourcetype 181
 PUNCH bookname.sourcetype 182
 PUNCH modulename.OBJ 178
 PUNCH phasename.PHASE 175

[illegible]

- migrate old product 323
 - backup libraries and history file 323
 - create new format type 326
 - create temp. history file and library 324
 - install new format tape 327
 - install old libraries 325
 - isolate product 325

O

- old members to new libraries 192
- option LINK
 - virtual I/O 85
- OPTION LINK, emulation of 223
- optional products
 - sizing 50

P

- panels 3, 6, 12, 13, 15, 16
 - keyword summary 6, 13, 16
 - application development 13
 - system administration 6
 - system operation 16
 - path summary 3, 12, 15
 - application development 12
 - system administration 3
 - system operation 15
- PDZAP, emulation of 219
- physical library migration
 - See backward migration
 - See forward migration
 - See VSE libraries transfer
- POWER 23
 - generation changes 23
 - startup
 - use of IESWAITT 319
- POWER files
 - tailoring 34
- PROMPT
 - VTAM startup option 318
- PSERV
 - DSPCH procedurename 185
 - DSPLY ALL 183
 - PUNCH procedurename 184
- PSERV, emulation of

- DSPCH 266
- PUNCH 264

R

- RESTORE 311
 - LIST DEVICE=FBA 140
 - LIST DEVICE=3330 140
 - LIST DEVICE=3340 140
 - LIST DEVICE=3350 140
 - NOVERIFY 137, 139
 - RESTDLI 311
 - RESTORE CL 139
 - RESTORE PL 139
 - RESTORE RL 139
 - RESTORE SL 139
 - RESTORE VSAM datasets 313
 - RESTSQL 312
- restore ICCF user libraries 334
- restore of Interactive Interface VSAM files 335
- RESTORE, emulation of 214
- ROUTER facility
 - how it worked 348
 - ROUTER replacement 350
- ROUTER replacement 350
 - sample EXEC for conversion 351
- RSERV
 - DSPCH modulename 178
 - DSPLY ALL 176
 - PUNCH xxx.ALL 177
- RSERV, emulation of
 - DSPCH 272
 - DSPLY 268
 - PUNCH 270

S

- scan CMS tape 66
- scan ICCF tape 60
- scan PL/SSL tape 63
- scan unstructured input tape 68
- scanner tape creation EXEC 346
- SETSDL procedure 57
- sizing optional products
 - sample session 50
- SNA
 - sample VTAM book 320

- software component comparison
 - VTAM 317
- SQL initialization 314
- SQLSTART 314
 - after database RESTORE 314
 - RESTORE archived copy 315
 - SQL COLDLOG 314
- SSERV
 - DSPCH sublib.bookname 182
 - DSPLY ALL 179
 - DSPLY sublib.bookname 180
 - PUNCH sublib.ALL 181
- SSERV, emulation of
 - DSPCH 278
 - DSPLY 274
 - PUNCH 276
- startup
 - VTAM 318
- SUBMIT procedure (ICCF) 72
- system files and libraries
 - sample layout SYSWK1 25
 - sample layout SYSWK2 26

T

- tailoring files 24
- tailoring ICCF DTSFILE 29
- tailoring libraries 24, 27
- tailoring libraries in BAM space 27
- tailoring libraries in VSAM managed space 28
- tailoring other files 42
- tailoring POWER files 34
- terminals
 - used by VTAM 318

V

- VIO 85
- virtual I/O 85
 - implications 86

- LINKGO CMS EXEC 91
- LINKGO ICCF CLIST 88
- LINKGO ICCF CLIST output 96
- usage of option CATAL 87
- VM/PASSTHRU 337
 - sample PASSTHRU EXEC 337
 - sample session using 338
- VSAM 308
 - BACKUP ALL 308
 - DEFINE SQL LOG dataset 309
 - DEFINE SQL VSAM datasets 310
 - DEFINE user catalog 309
 - RESTORE DL/I datasets 311
 - RESTORE SQL VSAM datasets 312
 - RESTORE VSAM datasets 313
- VSAM managed space
 - See libraries
- VSE libraries transfer 97, 189
 - description of DASD layouts 101, 102
 - layout SIPO31 101
 - layout SYSWK1 101
 - Stdlabels for SIPO31 102
 - Stdlabels for SIPO50 102
 - general librarian functions 98
 - mapping of libraries 103
 - Create Library Migration Table 104
 - Library Migration Graphic 103
 - Load Library Migration Table 105
 - mapping of old vs. new librarian functions 99
 - new librarian functions 100
 - old librarian functions 99
- VTAM
 - books 317
 - cross-domain 322
 - PROMPT option 318
 - sample local non-SNA 321
 - sample local SNA 320
 - startup book 317

This form may be used to communicate your views about this publication. They will be sent to the author's department for whatever review and action, if any, is deemed appropriate. Comments may be written in your own language; use of English is not required.

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

Note: *Copies of IBM publications are not stocked at the location to which this form is addressed. Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.*

Possible topics for comments are:

Clarity Accuracy Completeness Organization Coding Retrieval Legibility

If you wish a reply, give your name and mailing address:

What is your occupation? _____

Number of latest Newsletter associated with this publication: _____

Thank you for your cooperation. No postage stamp is necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments or you may mail directly to the address in the Edition Notice on the back of the title page).

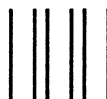
Note: Staples can cause problems with automated mail sorting equipment.
Please use pressure sensitive or other gummed tape to seal this form.

Reader's Comment Form

Fold And Tape

Please Do Not Staple

Fold And Tape



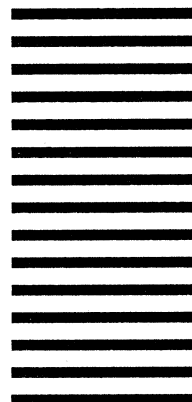
NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 40 ARMONK, N.Y.

POSTAGE WILL BE PAID BY ADDRESSEE:

International Business Machines Corporation
Department 6R1
180 Kost Road
Mechanicsburg, PA 17055



Fold And Tape

Please Do Not Staple

Fold And Tape



Cut or Fold Along Line

VSE/System Package Migration Volume 2 (File No. S370/4300-36) Printed in U.S.A. SC33-6179-1

This form may be used to communicate your views about this publication. They will be sent to the author's department for whatever review and action, if any, is deemed appropriate. Comments may be written in your own language; use of English is not required.

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

Note: *Copies of IBM publications are not stocked at the location to which this form is addressed. Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.*
Possible topics for comments are:

Clarity Accuracy Completeness Organization Coding Retrieval Legibility

If you wish a reply, give your name and mailing address:

What is your occupation? _____

Number of latest Newsletter associated with this publication: _____

Thank you for your cooperation. No postage stamp is necessary if mailed in the U.S.A.
(Elsewhere, an IBM office or representative will be happy to forward your comments or you may mail directly to the address in the Edition Notice on the back of the title page).

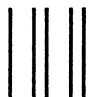
Note: Staples can cause problems with automated mail sorting equipment.
Please use pressure sensitive or other gummed tape to seal this form.

Reader's Comment Form

Fold And Tape

Please Do Not Staple

Fold And Tape



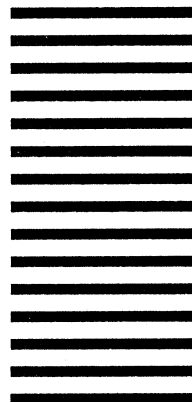
NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 40 ARMONK, N.Y.

POSTAGE WILL BE PAID BY ADDRESSEE:

International Business Machines Corporation
Department 6R1
180 Kost Road
Mechanicsburg, PA 17055



Fold And Tape

Please Do Not Staple

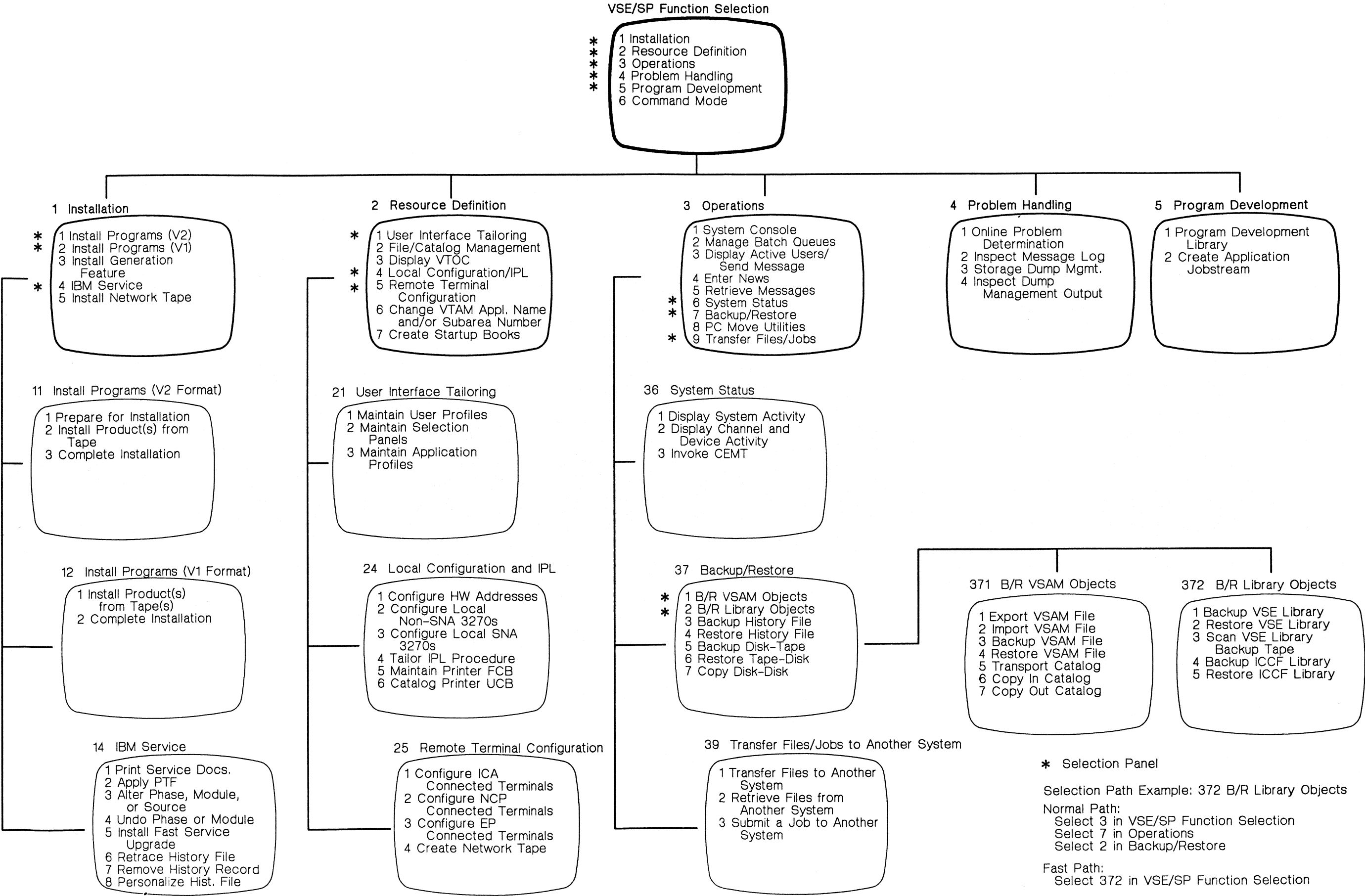
Fold And Tape



Cut or Fold Along Line

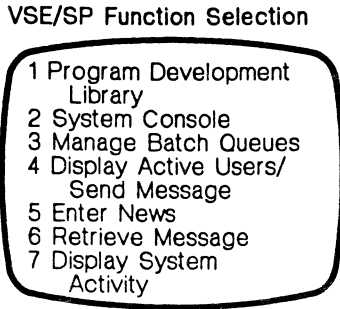
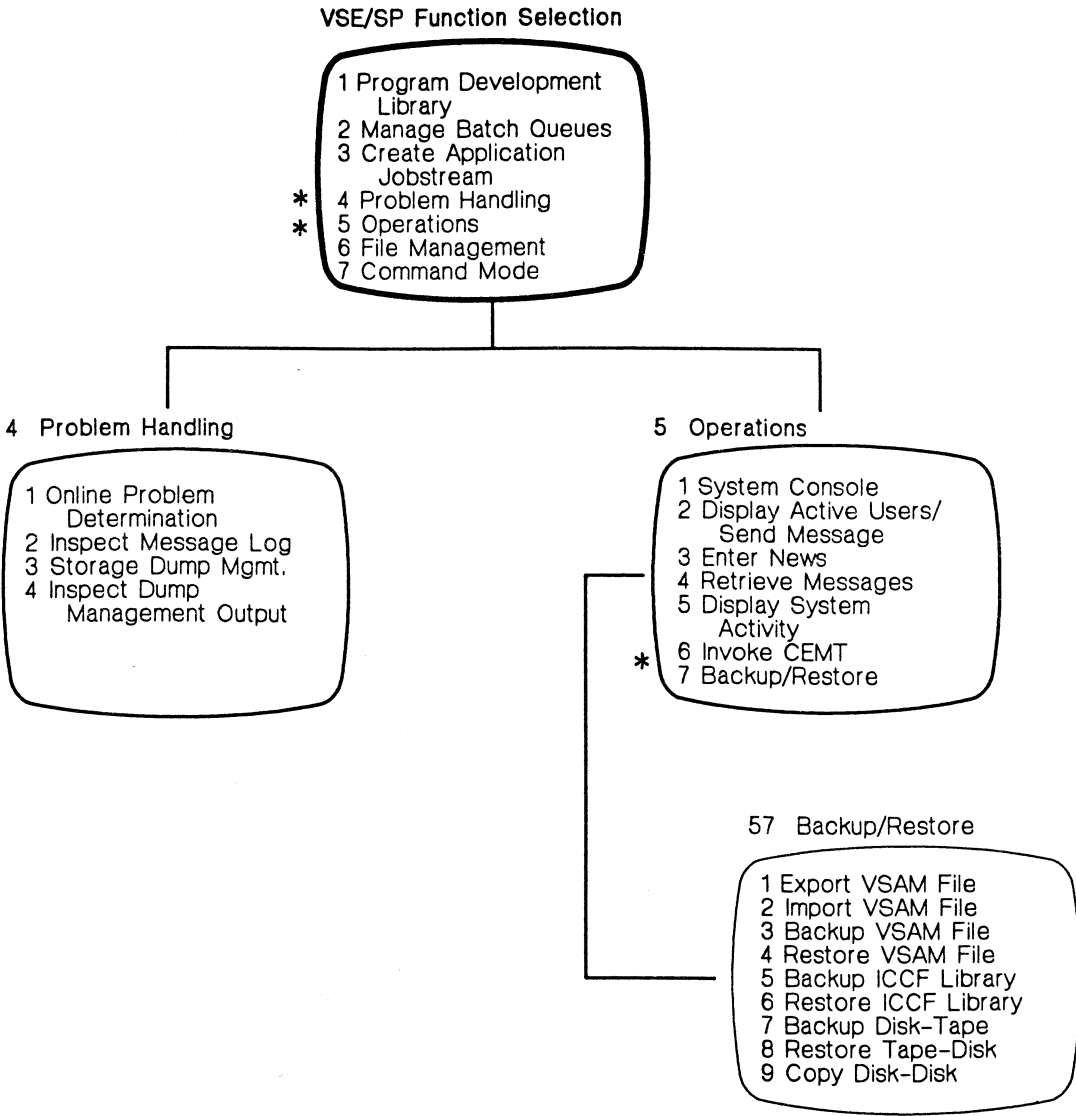
VSE/System Package Migration Volume 2 (File No. S370/4300-36) Printed in U.S.A. SC33-6179-1

Default Selection Panel Hierarchy for System Administrator



Default Selection Panel Hierarchy
for Programmer

Default Selection Panel for Operator



* Selection Panel

Selection Path Example: 57 Backup/Restore

Normal Path:
Select 5 in VSE/SP Function Selection
Select 7 in Operations

Fast Path:
Select 57 in VSE/SP Function Selection