CA Librarian®

Implementation Guide

r4.4

CA Technologies Product References

This document references the following CA Technologies products:

- CA Workload Automation CA 7 Edition® Workload Automation
- CA ACF2™ (CA ACF2)
- CA APCDOC™ Automated Job Documentation (CA APCDOC Automated Job Documentation)
- CA Earl™
- CA Librarian® for z/OS (CA Librarian)
- CA MetaCOBOL+® (CA MetaCOBOL+)
- CA Netman™
- CA Roscoe Interactive Environment® ([assign the caroscoe value for your book])
- CA Scheduler® Job Management (CA Scheduler JM)
- CA Top Secret® (CA Top Secret)
- CA Vollie™ (CA Vollie)

Documentation Changes

The following documentation updates have been made since the last release of this documentation:

- The Reserve/Release Test Programs appendix has been moved from the *Getting Started Guide* to the *Implementation Guide.*
- References to SAMPJCL have changed to CALJJCL.

Contact CA Technologies

Contact CA Support

For your convenience, CA Technologies provides one site where you can access the information that you need for your Home Office, Small Business, and Enterprise CA Technologies products. At http://ca.com/support, you can access the following resources:

- Online and telephone contact information for technical assistance and customer services
- Information about user communities and forums
- Product and documentation downloads
- CA Support policies and guidelines
- Other helpful resources appropriate for your product

Providing Feedback About Product Documentation

If you have comments or questions about CA Technologies product documentation, you can send a message to techpubs@ca.com.

To provide feedback about CA Technologies product documentation, complete our short customer survey which is available on the CA Support website at http://ca.com/docs.

Contents

Chapter 1: Introduction and Functional Overview 13

What LIB/CCF Is 13
LIB/CCF Request Cycle 14
Customizing LIB/CCF for Your Site 15
Tracking Capabilities 15
Reports 16
- Automatically Produced Reports 16
- Requested Reports 17
- Batch Change Request Status Report 18
- CA Earl Reports 19

Chapter 2: Administrator Functions in LIB/CCF 21

System Profile Table (Option 12.0) 22
- System Profile Table Panel Commands 37

User Definition Table (Option 12.1) 37
- User Definition Table Panel Fields 38
- User Definition Table Panel Commands 38

Manager Definition Table (Option 12.2) 39
- Manager Definition Table Panel Fields 39
- Manager Definition Table Panel Commands 40

Programmer Definition Table (Option 12.3) 41
- Programmer Definition Table Panel Fields 41
- Programmer Definition Table Panel Commands 42

Master File Definition Table (Option 12.4: AllFusion CA-Vollie Only) 42
- Master File Definition Table Panel Fields 43
- Master File Definition Table Panel Commands 46

Language Definition Table (Option 12.5) 46
- Language Definition Table Panel Fields 47
- Language Definition Table Panel Commands 48

Production Master File Definition Table (Option 12.6: AllFusion CA-Vollie Only) 49
- Production Master File Definition Table Panel Fields 49
- Production Master File Table Panel Commands 50

History Master File Definition Table (Option 12.7: AllFusion CA-Vollie Only) 50
- History Master File Definition Table Panel Fields 50
- History Master File Definition Table Panel Commands 51

ID Conversion Table (Option 12.8) 52

CA Roscoe Key Conversion Table 52
ID Conversion Table Panel Fields 53
ID Conversion Table Panel Commands 53
ISPF SYSUID Conversion Table 53
CA Vollie OPNAME Conversion Table 54
Change Request Reassignment (Option 12.9) 55
Change Request Reassignment Panel Fields 56
Change Request Reassignment Panel Commands 56
Master File Information Table (Option 12.10: AllFusion CA-Vollie Only) 56
Master File Information Table Panel Fields 57
Master File Information Table Panel Commands 58
Library Chain Definition (Option 12.11: Except AllFusion CA-Vollie) 58
LCDF Selection List Processing 58
Create a Library Chain 60
Chain Description Panel 61
Chain Structure Panel 61
Chain Attributes Panel 63
Associated Files Panel 66
Library Definition Panels LCDF 67
Modify a Library Chain 73
Modify the Library Definition 73
Deactivate a Library Chain 74
Delete Deactivated Chains 75
Library Chain Definition Report (Option 8.9) 75
Group Definition Table (Option 12.12 Except AllFusion CA-Vollie) 76
Group Definition Table Panel Fields 77
Group Definition Table Panel Commands 77

Chapter 3: LIB/CCF Batch Utilities for CA Roscoe and ISPF(TSO) 79

Change Request Status Report ($CCFB101) 79
Parameters 80
Batch Logout Utility ($CCFB102) 81
Condition Codes 81
Parameters 82
SYSIN Control Statements 82
Examples 84
Batch Login Utility ($CCFB109) 84
Condition Codes 85
Group Processing Activity Report 85
ISPF Execution 88
CA Roscoe Execution 90

Batch Change Request Initiation ($CCFB103) 92
Parameters 92
SYSIN Control Statements 92
Processing Verbs 94
Example 95
Executing $CCFB103 from Another Program 96
Batch Change Request Close ($CCFB104) 97
Parameters 97
SYSIN Control Statements 98
Processing Verb 98
Example 99
Executing $CCFB104 from Another Program 100
LIBRSCAN 100
LIBRCOPY 101

Chapter 4: LIB/CCF Batch Utilities for CA Vollie 103

Change Request Status Report 103
Parameters 104
Batch Logout Utility 105
Parameters 106
Batch Change Request Initiation 108
Parameters 109
Executing CCFB103 from Another Program 112
Batch Change Request Close 113
Parameters 113
Executing CCFB104 from Another Program 115
LIBDSCAN 116
LIBDCOPY 116

Chapter 5: LIB/CCF Model System 117

Basic Function 117
Change Request 118
History Member 118
Test Object 119
Link-Edit JCL Member 119
System Master File 120
Model System 122
Model System Defaults 122
Model System Requirements 123
Model System Demo Script 123
Recommendations for Setting Up Your System 124

Security Protection 124
Source to Load Module Verification 125
User Exit Points for Customizing LIB/CCF 128
Suggested Procedure for Emergency Changes 128
Multiple Programmers Assigned to One Member 129

Chapter 6: Exception Handling 133

Reassigning a Change Request 133
Rerunning a Failed Batch Job 133
For VSE/ESA 134
For VM/ESA 134
For z/OS and OS/390 135
Possible Logout Problems 136
Member is Already on the Test Master File 136
Member is Already Logged Out 136
Control Group Rejects the Logout Request 137
Member Logout Limit is Exceeded 137
Cancelling a Logout 137
Possible Login Problems 137
Member is Not on the Test Master File 137
Assembly or Compile Fails 138
There are Multiple Logouts 138
Control Group Rejects the Login Request 141
Multiple Administrator Access (z/OS and OS/390 and VM/ESA Only) 142

Chapter 7: Implementing Quality Assurance Procedures 145

Q/A Process in LIB/CCF 145
Method One: Defining a Production-Test Promotion Path 147
Method Two: Defining Q/A Libraries in Promotion Paths 148
Implementing Method Two in CA Vollie 148
Step 1: Define the Library Promotion Paths 149
Step 2: Complete the Q/A Worksheet 149
Step 3: Interpret the Q/A Worksheet 153
Step 4: Make the Appropriate Table Entries 158
Verify the Q/A Configuration 163

Chapter 8: Scheduling z/OS and OS/390 Batch Jobs in LIB/CCF 165

Implementation Considerations 166
For CA 7 167
For CA Scheduler 168

Operation 169

Chapter 9: DB2 for z/OS and OS/390 Support in LIB/CCF 171

Implementation Considerations 173
For ISPF 173
For CA Roscoe 174
DB2 for z/OS and OS/390 LIB/CCF Audit Trail 175
Creating the B2 for z/OS and OS/390 Table 176
Binding the Plans 178
Updating the CALIBCCF.LIBCCFTA Table 180
Maintaining the CALIBCCF.LIBCCFTA Table 182

Chapter 10: LIB/CCF-CA Earl Interface 183

Sample Reports 183
LJEARLCR and LJEARLCA 184
LJEARLSY 185
z/OS and OS/390 Report Generation 186
VSE/ESA Report Generation 188
VM/ESA Report Generation 191

Chapter 11: LIB/CCF-CA Netman Interface 193

Installation 194
Operations 196

Chapter 12: Customizing Your System: LIB/CCF-CA Roscoe 199

User Exits 200
Sample User Exits 206
JCL Skeletons 207
Manipulating JCL Processing 217
Link-Edit Considerations 224
Customizing Panels and Reports 226
Execution 228
Identifying Panels 228
PF Keys and the Jump Command 228
Job Card Specification 229
System Error Processing 230

Chapter 13: Customizing Your System: LIB/CCF-ISPF(TSO) 231

User Exits 231

Sample User Exits 241
Sample User Exit $CCFU007 242
JCL Skeletons 244
Manipulating JCL Processing 254
Link-Edit Considerations 256
Customizing Panels and Reports 258
Execution 259
PF Keys and the Jump Command 260
Job Card Specification 260
System Error Processing 261

Chapter 14: Customizing Your System: LIB/CCF-ISPF(VM/ESA) 263

User Exits 263
Sample User Exits 273
JCL and EXEC Skeletons 275
$CCxJ000 Skeleton 276
$CCxJ001 Skeleton 276
$CCxJ002 Skeleton 277
$CCxJ003 Skeleton 278
$CCxJ004 Skeleton 279
$CCxJ005 Skeleton 279
$CCxJ006 Skeleton 280
$CCxJ007 Skeleton 280
$CCxJ009 Skeleton 281
$CCxJ010 Skeleton 281
$CCxJ011 Skeleton 282
$CCxJ012 Skeleton 282
$CCxJT10 Skeleton 282
$CCxJT11 Skeleton 283
$CCxJT12 Skeleton 283
$CCFJ030 Skeleton 283
$CCFJ031 Skeleton 284
$CCFJ032 Skeleton 285
$CCxJ100 Skeleton 285
$CCxJ101 Skeleton 286
$CCFJ103 Skeleton 286
$CCCJ200 Skeleton 287
$CCCJ201 Skeleton 288
$CCFJ200 Skeleton 289
$CCFJ201 Skeleton 289
Link-Edit Considerations 290

Customizing Panels and Reports 292
Execution 293
PF Keys and the Jump Command 294
Job Card Specification 295
System Error Processing 296

Chapter 15: Customizing Your System: LIB/CCF CA Vollie 297

User Exits 297
Sample User Exits 305
JCL Skeletons 307
$CCVJ000 Skeleton 307
LIBR--$CCVJ001 Skeleton 308
$CCVJ002 Skeleton 308
$CCVJ003 Skeleton 308
$CCVJ004 Skeleton 309
$CCVJ005 Skeleton 309
$CCVJ006 Skeleton 309
$CCVJ007 Skeleton 310
$CCVJ008 Skeleton 310
$CCVJ009 Skeleton 310
$CCVJ010 Skeleton 310
$CCVJ011 Skeleton 311
$CCVJ012 Skeleton 311
$CCVJT10 Skeleton 311
$CCVJT11 Skeleton 312
$CCVJT12 Skeleton 312
$CCVJ090 Skeleton 313
$CCVJ100 Skeleton 313
$CCVJ200 Skeleton 314
Manipulating JCL Processing 314
Link-Edit Considerations 315
Customizing Panels 316
Customizing Reports 318
Module History Report 318
Change Request Report 319
Execution 319
PF Keys and Commands 319
Job Card Specification 320
System Error Processing 320

Appendix A: Extension Variables 323

Appendix B: Troubleshooting 333

Diagnostic Procedures 334
Collecting Diagnostic Data 335
Interpreting Diagnostic Data 335
Accessing the Online Client Support System 336
- StarTCC 336
- Requirements for Using StarTCC 336
- StarTCC Security 337
- Accessing StarTCC 337

Accessing the Technical Support Phone Services Directory 338
CA-TLC: Total License Care 338
Contacting Technical Support 339
Product Versions and Maintenance 339
Requesting Enhancements 340

Appendix C: Reserve/Release Test Programs 341

How the Programs Work 341
Testing Reserve/Release From z/OS and OS/390 342
Testing Reserve/Release From VSE/ESA 343
Testing Reserve/Release From VM/ESA and z/VM 346

Glossary 347

Index 351

Chapter 1: Introduction and Functional Overview

This guide provides the LIB/CCF administrator with information on setting up and maintaining the Change Control Facility. The distribution of this guide must be restricted because it contains discussions of the management code (MCD) security feature. See the *Security Administration Guide* for a thorough understanding of the workings of the access control interface and the management code (MCD).

This section contains the following topics:

What LIB/CCF Is (see page 13)
LIB/CCF Request Cycle (see page 14)
Customizing LIB/CCF for Your Site (see page 15)
Tracking Capabilities (see page 15)
Reports (see page 16)

What LIB/CCF Is

The Change Control Facility (LIB/CCF) is an interactive dialog-based application that provides CA Librarian users with a comprehensive change control methodology. It includes complete tracking and control for source program development in online environments.

LIB/CCF is based on a simple, practical philosophy of program change control. The basis for this philosophy is that there is a natural separation between the production environment and the test environment. To manage the movement of program source between these environments, there must be a control point.

LIB/CCF is an application specifically designed to define and manage that control point, taking full advantage of all of the control, auditing, and recovery capabilities that are built into CA Librarian. LIB/CCF extends those capabilities by providing managers with systematic implementation procedures and a full range of reporting facilities for managing and monitoring the development cycle.

LIB/CCF provides a way for the user to enter change requests, the programming manager to review and assign change requests to a programmer, and to track change requests with a series of reports through the entire process of updating, testing, and linking the programs to modify.

LIB/CCF Request Cycle

You can configure LIB/CCF several ways to carry through a change request from beginning to end. This section describes the way that things happen in the model system.

The following sample displays the options available from the ISPF LIB/CCF primary options menu.

```
           CA-LIBRARIAN CHANGE CONTROL FACILITY
OPTION===>
0  CCF PARMS        - Specify user parameters
1  OPEN             - Open a change request
2  ASSIGN           - Assign, reassign, reject a change request
3  DISPLAY          - Display assigned change requests
4  LOGOUT           - Logout a module to test
5  LOGIN            - Login a module to production
6  LINK             - Initiate a production linkedit or bind
7  CLOSE            - Close a change request
8  REPORTS          - Print or display LIB/CCF reports
9  MOVEMENT STATUS  - Display, process, reject LOGOUT/IN requests
10 LINK STATUS      - Display, process, reject link/bind requests
11 JOB STATUS       - Display, process, resubmit pending jobs
12 ADMIN SERVICES   - LIB/CCF administrator services
 T TUTORIAL         - Display information about LIB/CCF
 X EXIT             - Terminate LIB/CCF
```

First, the user of an application uses Option 1 of LIB/CCF to open the change request online. The change request enters the queue of the programming manager in charge of that application.

The manager uses Option 2 to review the request, add comments if necessary, and assign the request to one of the programmers defined to LIB/CCF as reporting to him.

The programmer to whom the request is assigned views it on Option 3, logs out the source members from the production environment to test using Option 4, and makes the requested changes on the test master file.

When the changes are completed and tested, the programmer uses Option 5 to request that the control group log in the source members to the production master file. Once this is done, the programmer uses Option 6 to request that the control group link the members. Finally, the programmer uses Option 7 to close the change request.

Customizing LIB/CCF for Your Site

The scenario just described is not the only way that LIB/CCF lets you do things. The LIB/CCF Administrator, using Option 12 to alter the LIB/CCF tables, can tailor LIB/CCF in one of many different configurations to meet the needs of your site.

The control group can logout the source member from the production to the test environment instead of the programmer (the TMOVEMENT keyword of the administrator's Option 12.0 controls this option). The transfer can take place either online or in batch (keywords TMOVEMENT and CMOVEMENT of Option 12.0).

The programmer or the control group can login from test to production (keyword PMOVEMENT of Option 12.0), online or in batch (keywords PMOVEMENT AND CMOVEMENT of Option 12.0). You cannot do batch movement in a VM/ESA batch machine.

The application system link has these same options (LKED and CLINK keywords of Option 12.0). Online link is not available in LIB/CCF-ISPF(VM/ESA) or LIB/CCF-AllFusion™ CA-Vollie®.

For LIB/CMS-ISPF(VM/ESA), additional information to be supplied in Option 0 on the JOBCARD panel is spool machine ID, spool tag ID, and the spool machine type. Programmers have an opportunity to change the spool machine type before batch-only logouts (Option 4) and any logins (Option 5). When a request is processed, a Machine Type specification panel displays. The rest of the job card information displays to the control group, where they can further modify it if necessary. These panels default to data entered in Option 0.

Tracking Capabilities

When a function is performed in batch mode, a job submission record tracks the job. The functions for which LIB/CCF creates and manages a job submission record include transfer of a member between the test and production environments, compiles and assemblies, and link editing members to produce new production executable code.

Test to production movement and compiles are triggered in Option 5 if the programmer performs these functions, or Option 9 if the control group is responsible.

Linking is initiated in Option 6. If the control group performs the link, this group uses Option 10 to process the programmer's request.

When the job completes with a return code of 0, the JSR is automatically removed. If the job fails for any reason, either the programmer or the control group (whichever the site designates) can resubmit the job if necessary (Option 11) or process the JSR to remove it (also Option 11). See the "Exception Handling" chapter for more details on exception handling.

When the control group is to perform the transfer from test back into production, the login function places a member movement record in the login/logout status display.

Viewing this display through Option 9, a member of the control group can process the request or reject it. If the request is rejected, LIB/CCF marks the member movement record rejected. The programmer can either reprocess the login request or delete the entry and resubmit the logout request.

Similarly, LIB/CCF places an entry in the system link request display when the control group is defined as controlling this function. A control group member can either process or reject the entry. To notify the programmer of the rejection, the rejected entry is marked as such and left in the display. Using Option 10, the programmer deletes the rejected link request.

Reports

Certain LIB/CCF reports are produced automatically at various stages in the modification cycle. You can request other reports through Option 8. Batch summary AllFusion reports providing a comprehensive overview of ongoing programming activities are also available.

For LIB/CMS-ISPF(VM/ESA), a PRINT function is available from Option 0 to display a Report Print Specification panel. You can make entries on this panel to direct where print requests are forwarded and to record other print information. You can either spool or direct reports to a user-defined VM/ESA file. This panel redisplays for modification, if necessary, whenever a print request is made.

Automatically Produced Reports

Site options control what reports are produced automatically in the course of LIB/CCF operations. In the model system, four reports are automatically produced, but the site administrator can turn off any or all of them using Option 12.0 to alter the System Table.

In LIB/CCF-ISPF(VM/ESA), the Report Print Specification panel displays before any of the following reports are automatically produced so that you can modify the specifications.

- A Change Request Report is directed to the requestor when the change request is opened through Option 1. The AUTOREPT1 keyword of Option 12.0 controls whether this report is automatically produced.
- An Assignment Notification Report is also directed to the requestor when the applications manager uses Option 2 to assign the change request. The AUTOREPT2 keyword of Option 12.0 controls whether this report is automatically produced.

- A Module History Report is directed to the programmer when a member is logged in using Option 5. The AUTOREPT5 keyword of Option 12.0 controls whether this report is automatically produced.
- When the programmer uses Option 7 to close the change request, a complete Change Request Report shows the original change request, a report of all activity (logouts, logins, links, and so on), and a close section describing the details of the changes made to complete the request. The AUTOREPT7 keyword of Option 12.0 controls whether this report is automatically produced. When produced, three copies of the Change Request Report are directed to the programmer. They can be distributed as the site deems best. Ordinarily, one is for the requestor, one for the applications manager, and one for the programmer.

Requested Reports

Site option also controls which reports can be specially requested through Option 8. These reports are available to the application manager, the programmer, the members of the control group, or some combination of these, depending on what the LIB/CCF administrator specified using Option 12.0. You can either print or view the reports online.

The following is a list of the available reports, who can request them, and their contents:

- Unassigned Requests Report—(Option 8.1) Displays unassigned requests. The applications manager can request this report for his own application system.
- Assigned Requests Report—(Option 8.2) Displays assigned requests. A programmer or applications manager can request the report. The programmer sees all the requests assigned to him. The applications manager sees all the requests assigned to all the programmers who report to him.
- Module Logout Report—(Option 8.3) Displays information about all members logged out to the requesting programmer. A programmer, an application manager, or a member of the control group can request this report. The applications manager can choose to see this information about logged out members for all programmers who report to him. The control group can request the report for one programmer or a full report for all.
- Module History Report—(Option 8.4) Displays a member's history information. The LIB/CCF system profile table (Option 12.0) contains two keywords (CNTLPRINT and HISTREPT) that determine who can request this report.
- Login/Logout Request Status Report—(Option 8.5) Displays outstanding login and logout requests from the programmer requesting the report. A programmer, manager, or control group member can request this report.
- Link Edit/Bind Request Status Report—(Option 8.6) Displays a list of all requests for system links and binds that were submitted to the control group for processing. A programmer, manager, or control group member can request this report.

- Pending Job Status Report—(Option 8.7) Displays a list of all jobs submitted on behalf of the user requesting the report. These can include jobs to transfer members between the test and production environments or jobs submitted to link edit a new production application. A programmer, manager, or control group member can request this report.
- Change Request Report—(Option 8.8) Displays a complete or partial Change Request report. A complete Change Request Report shows the original change request, a report of all activity (logouts, logins, links, and so on), and a close section describing the details of the changes made to complete the request. Only the programmer to whom the change request is assigned and his manager can request this report. Additionally, anyone or the control group can be authorized to request the report, based on the CNTLPRINT and CRREPT keywords of Option 12.0.
- Library Chain Definition Report—(Option 8.9) This option generates several different formats of Chain and Library Definition Reports. Only the CCF System Administrator can request this report. This report is not available in LIB/CCF CA Vollie.
- DB2 for z/OS and OS/390 Reports—(Option 8.10) This option lets you select several different formats of DB2 for z/OS and OS/390 reports. This report is available only in LIB/CCF CA Roscoe Interactive Environment (CA Roscoe) and TSO.

Batch Change Request Status Report

The Change Request Status Report program can be used to print comprehensive summary reports. Depending on the options selected, this report can be produced all change requests, all open change requests, all unassigned change requests, or all closed change requests.

The report will list the change requests for either a single programmer, a manager, or an application system.

Optionally the Batch Change Request Status report will print the analysis section for closed change requests, as well as the status of logouts, links, moves, and batch jobs associated with each selected change request. See your LIB/CCF administrator for details on the availability and use of this report utility.

More information:

LIB/CCF Batch Utilities for CA Roscoe and ISPF(TSO) (see page 79)
LIB/CCF Batch Utilities for CA Vollie (see page 103)

CA Earl Reports

LIB/CCF is distributed with a [set the earl variable for your book] (Easy Access Report Generation) interface. [set the earl variable for your book] is a report generation system that uses a high-level programming language to produce customized reports. The interface provides access to information collected by LIB/CCF.

Chapter 2: Administrator Functions in LIB/CCF

Administration of LIB/CCF is managed online by a series of function panels found under Option 12. Through these panels, the LIB/CCF administrator can:

- Maintain the LIB/CCF system profile.
- Identify authorized requestors, application managers, programmers, and groups to the system.
- Define and control access to test, Q/A, reject, and production source libraries (CA Librarian master files), object libraries, and executable libraries.
- Define LIB/CCF system and history master files.
- Reassign a change request.

The LIB/CCF administrator must be defined to the system. In the CA Roscoe environment, the administrator is the CA Roscoe key that contains the LIB/CCF RPF programs. Additional administrators can be defined by the ADMINKEY keyword of Option 12.0 (PROFILE). Only the administrator has access to Option 12, the option to view these function panels, on the main menu. Option 12 does not display to other users.

In the ISPF(TSO) and ISPF(VM/ESA) environments, the administrator is the TSO ID or VM/ESA ID that is designated in the $CCFGEN macro by the keyword ADMINID. Additional administrators can be defined by the ADMINID keyword of Option 12.0 (PROFILE). While Option 12 displays to all in LIB/CCF-ISPF, only the administrators can select this option.

In the CA Vollie environment, the administrator is the owner of the OPIDENT specified in the CCFA member of the VOICE library where CCF was installed.

After the programs are installed and all the required files are allocated, the LIB/CCF administrator tailors the LIB/CCF system to meet site requirements using the function panels of Option 12. These panels, Options 12.0 through 12.11, are table processors, as follows:

- 12.0—System Profile Table
- 12.1—User Definition Table
- 12.2—Manager Definition Table
- 12.3—Programmer Definition Table
- 12.4—Master File Definition Table (CA Vollie only)
- 12.5—Language Definition Table

- 12.6—Production Master File Table (CA Vollie only)
- 12.7—History Master File Table (CA Vollie only)
- 12.8—Identification Key Conversion Table
- 12.9—Reassign Change Requests
- 12.10—Master File Information Table (CA Vollie Only)
- 12.11—Library Chain Definition (Except CA Vollie)
- 12.12—Group Definition Table (Except CA Vollie)

In addition to these administrator functions, an option to produce a Library Chain Definition Report (Option 8.9) is available to the administrator (except for CA Vollie).

Each of the following panels bears a number of keywords for which the LIB/CCF administrator must enter values. There is also a command line on each panel. The following sections describe the purpose of the panel in general terms, list the keywords and the options that you can specify for them, and then list the LIB/CCF commands that you can enter on the command line. For the most part, the available keywords are common to all online environments LIB/CCF supports. Minor variations are noted where necessary.

This section contains the following topics:

System Profile Table (Option 12.0) (see page 22)
User Definition Table (Option 12.1) (see page 37)
Manager Definition Table (Option 12.2) (see page 39)
Programmer Definition Table (Option 12.3) (see page 41)
Master File Definition Table (Option 12.4: AllFusion CA-Vollie Only) (see page 42)
Language Definition Table (Option 12.5) (see page 46)
Production Master File Definition Table (Option 12.6: AllFusion CA-Vollie Only) (see page 49)
History Master File Definition Table (Option 12.7: AllFusion CA-Vollie Only) (see page 50)
ID Conversion Table (Option 12.8) (see page 52)
Change Request Reassignment (Option 12.9) (see page 55)
Master File Information Table (Option 12.10: AllFusion CA-Vollie Only) (see page 56)
Library Chain Definition (Option 12.11: Except AllFusion CA-Vollie) (see page 58)
Group Definition Table (Option 12.12 Except AllFusion CA-Vollie) (see page 76)

System Profile Table (Option 12.0)

The System Profile Table defines the system options for the site. Here, the administrator specifies:

- Whether multiple logouts can take place
- What reports are automatically produced

- How system links and member movements are performed and by whom
- Who is authorized to request what reports
- Members of the control group

- User exits to invoke
- Various other details of LIB/CCF execution

The following sample screen shows a sample ISPF (TSO) System Profile Table display.

```
 TUE mm/dd/yy ----------- SYSTEM PROFILE TABLE ----------- ROW 1 OF 145
                                                              07.26.03
COMMAND ===>                                              SCROLL ===> PAGE
            KEYWORD     VALUE
            ADMINID     SMITH
            ALLOWMULT   YES
            AUTOREPT1   NO
            AUTOREPT2   NO
            AUTOREPT5   NO
            AUTOREPT7   YES
            CLINK       ONLINE
            CLOSE       ALL
            CMOVEMENT   ONLINE
            CNTLID      BROWN
            CNTLPRINT   YES
            CRREPT      ALL
            DB2PKG      YES
            EMERID      BROWN
            EXIT14      YES
            GROUPLOGIN  PROCESS
            HISTREPT    YES
            HISTUPD     YES
            JSQUEUE     CNTL
            JSREDIT     NO
            LDISPLAY    NO
            LKED        CNTL
            LNKNTEGRTY  YES
            LOGLIMIT    050
            MDELETE     YES
            MDISPLAY    YES
            NOTIFY      YES
            PMOVEMENT   CNTL
            PRINT       ONLINE
            SCHEDULER   CA-Scheduler
            SYSOUT      T
            TMOVEMENT   ONLINE
            UNIT        SYSDA
```

The keywords for the panel are:

ADMINID

(TSO and VM/ESA.) Designates the TSO or VM/ESA ID of a LIB/CCF administrator and the ID defined by the ADMIN option of the $CCFGEN installation macro. Specify this keyword once for each additional administrator. The ADMINID keyword provides the same authorization for administrator functions as the ADMIN option of the $CCFGEN macro.

ADMINKEY

(CA Roscoe only) Designates the LIB/CCF key of each LIB/CCF administrator, and the CA Roscoe key where LIB/CCF is installed. Specify this keyword once for each administrator. The LIB/CCF key is the first eight characters of the CA Roscoe key or, if converted, the LIB/CCF key as defined by Option 12.8 (CA Roscoe key conversion table). The ADMINKEY keyword provides the same authorization for administrator functions as the key where LIB/CCF is installed.

ALLOWMULT

Indicates whether more than one programmer can log out the same member at the same time. The options are YES and NO. For information on the implications of ALLOWMULT, see the Multiple Programmers Assigned to One Member section in the "LIB/CCF Model System" chapter.

AUTOREPT1

Indicates whether the Change Request Report is automatically printed when the Change Request is opened from Option 1. The options are YES and NO.

AUTOREPT2

Indicates whether the Assignment Notification Report is automatically printed when the Change Request is assigned from Option 2. The options are YES and NO.

AUTOREPT5

Indicates whether the Member History Report is automatically printed when the member is logged in from Option 5. The options are YES and NO.

AUTOREPT7

Indicates whether the Change Request Report is automatically printed when the change request is closed from Option 7. The options are YES and NO.

BATCHCMS

(VM/ESA only.) Identifies the machine IDs where batch VM/ESA jobs can be spooled. Multiple entries are allowed. A value of ANY spools jobs to any machine ID. This keyword is optional. If omitted, users cannot submit batch VM/ESA jobs from LIB/CCF.

BATCHMVS

(VM/ESA only.) Identifies the machine IDs where batch z/OS and OS/390 jobs can be spooled. Multiple entries are allowed. A value of ANY spools jobs to any machine ID. This keyword is optional. If omitted, users cannot submit batch z/OS and OS/390 jobs from LIB/CCF.

BATCHVSE

(VM/ESA only.) Identifies the machine IDs where batch VSE/ESA jobs can be spooled. Multiple entries are allowed. A value of ANY spools jobs to any machine ID. This keyword is optional. If omitted, users cannot submit batch VSE/ESA jobs from LIB/CCF.

CA7JCLID

(CA Roscoe and TSO.) Required if CA 7 Workload Automation is specified for the SCHEDULER keyword. A three-digit number that must match the three-digit number specified by the CA 7 online JCL *//JCLnnn* statement that identifies the system master file. The default is 999.

CA7OPERID

(CA Roscoe and TSO.) Required if Unicenter CA-7 is specified for the SCHEDULER keyword. The one- to six-character operator ID for the Unicenter CA-7 logon. The default is CCF

CA7PASS

(CA Roscoe and TSO.) Optional, for use with Unicenter CA-7. LIB/CCF supports a one- to eight-character Unicenter CA-7 logon password. There is no default.

CLINK

(CA Roscoe and TSO.) Defines how the control group links and DB2 for z/OS and OS/390 binds the production application system. The options are:

BATCH

Immediately submits a batch job to execute the link-edit or bind.

ONLINE

Accomplishes the link-edit or bind interactively.

Note: If LKED does not specify CNTL, this keyword is ignored.

CLOSE

Defines who can close a change request.

ALL

Indicates that the programmers assigned to the change request, the programmer's manager, and control group members can close the change request.

PGMR

Indicates that the programmers assigned to a change request can close the change request.

CNTL

Indicates that the control group members can close any change request.

MANAGER

Indicates that the manager of the application that the change request was opened for can close the change request.

CMOVEMENT

Defines how members are moved to and from the production master file if the control group (anyone defined by the CNTLID or CNTLKEY keywords) is responsible for the movement (that is, the TMOVEMENT and PMOVEMENT keywords specified CNTL). If quality assurance (Q/A) libraries are defined in the library promotion path, this keyword also affects the movement of members to and from these libraries. The options are:

BATCH

Immediately submits a batch job that performs the copy to or from production. If the movement is to production, the job also deletes the member from the test master, and performs optional, subsequent processing. See note below. Batch movement cannot be done in a VM/ESA batch machine.

ONLINE

Immediately copies the member to or from production. If the movement is to production, also deletes the member from the test master. It then submits a batch job to perform optional subsequent processing. See the following notes.

Either all archive levels or the current level of the member are copied. In z/OS and OS/390 and VM/ESA, the SOURCE MOVE specification in the Library Chain Definition Function determines this. In LIB/CCF-AllFusion CA-Vollie, the COPY specification of the Master File Definition Table determines this At login time, LIB/CCF uses the language assigned to the member during its initial logout to determine what JCL skeleton (if any) performs “subsequent processing” The LANG-to-JCL-skeleton relationship is defined in Option 12.5. For source members, “subsequent processing.” can include either compiling from the production master file or copying the test object module to the production object library.

In z/OS and OS/390 and VM/ESA, the OP SYS MOVE specification in the Library Chain Definition Function determines this.

In LIB/CCF-AllFusion CA-Vollie, the presence of a TEST RELOCATABLE LIBRARY in the Master File Definition Table determines this.

CNTLID

(TSO and VM/ESA.) Designates the TSO or VM/ESA ID of a control group member. Specify this keyword once for each member of the control group. This keyword is optional. If omitted, there is no control group.

CNTLKEY

(CA Roscoe and CA Vollie.) Designates the LIB/CCF key of each control group member. Use this keyword once for each person in the control group. The control group can consist of any number of LIB/CCF keys. This keyword is optional. If omitted, there is no control group. The control key specification cannot exceed eight characters.

- For CA Roscoe, the LIB/CCF key is either the first eight characters of the CA Roscoe key or the LIB/CCF key as defined by Option 12.8 (CA Roscoe key conversion table).
- For CA Vollie, the LIB/CCF key is either the first eight characters of the CICS operator name or the LIB/CCF key defined by Option 12.8 (OPNAME Conversion Table).

CNTLPRINT

Indicates whether the control group is allowed to print or view Change Request and Module History reports. The options are YES and NO.

CRREPT

Indicates whether LIB/CCF ID validation is performed when the Change Request report (Option 8.8) is requested. The options are:

RESTRICT

Only the last person to whom the change request was assigned and that person's manager can view the change request report.

ALL

Anyone can view the change request report.

DATEFORM

(CA Vollie only.) Indicates the operating system date format. The options are:

AMERICAN

mm/dd/yy

EUROPEAN

dd/mm/yy

DB2PKG

Indicates whether DB2 release 2.3 or later is supported (specifically, DB2 packages).

YES

Indicates that DB2 release 2.3 is used at your site.

NO

Indicates that DB2 release 2.3 is not supported.

EMERID

(TSO and VM/ESA.) Designates the TSO or VM/ESA ID of each emergency group member. Specify this keyword once for each member of the emergency group. This keyword is optional. If omitted, there is no emergency ID.

Note: Even if ALLOWMULT is NO, the IDs entered here are allowed to do multiple logouts. This enables emergency changes to members that are already logged out. See the Suggested Procedure for Emergency Changes section in the "LIB/CCF Model System" chapter for details.

EMERKEY

(CA Roscoe and CA Vollie.) Designates the LIB/CCF key of each emergency group member. Specify this keyword once for each member of the emergency group. This keyword is optional. If omitted, there is no emergency key.

- For CA Roscoe, the LIB/CCF key is either the first eight characters of the CA Roscoe key or the LIB/CCF key as defined by Option 12.8 (CA Roscoe key conversion table).
- For CA Vollie, the LIB/CCF key is either the first eight characters of the CICS operator name or the LIB/CCF key defined by Option 12.8 (OPNAME Conversion Table).

Note: Even if ALLOWMULT is NO, the IDs entered here are allowed to do multiple logouts. This enables emergency changes to members that are already logged out. See the Suggested Procedure for Emergency Changes section in the "LIB/CCF Model System" chapter for details

EXITn

(Except CA Vollie.) Indicates whether LIB/CCF invokes the specified user exit. The options are YES and NO. Since NO is the default, this keyword need not be present if you are not using the exit.

- For ISPF, *n* can be a number from 0 to 15 (excluding 11).
- For CA Roscoe, *n* can be 2, 4, 5, 6, 7, 8, 9, 10, or 12. This parameter affects only batch logins performed by the $CCFB109 batch login utility.

FORMAT

(CA Roscoe only.) Indicates the printed format for CCF reports. The options are:

MIXED

Both upper and lower case.

UPCASE

Upper case letters only.

GROUPLOGIN

(Except CA Vollie.) Indicates the level of functionality for group login processing. The options are:

PROCESS

Specifies that move and compile functions must be performed together.

MOVE

Specifies that move and compile functions can optionally be performed separately, and members can be moved in a separate step before compiling

HISTREPT

Indicates whether LIB/CCF ID validation is performed when the History Report is requested. The options are:

YES

Specifies that only the last person to update the history and that person's manager can view the history report.

NO

Specifies that anyone can view the history report.

HISTUPD

Indicates whether the programmer can modify member history information (with the Option 5 HISTORY command) for a member that is currently logged out. The options are:

YES

(Default) Specifies that the programmer can issue the HISTORY command from Option 5 and modify the member's system, description, language, and abstract information.

NO

Specifies that the programmer cannot issue the HISTORY command.

JSQUEUE

Indicates whether the programmer or the control group can process the pending job activity from option 11 (JOB STATUS) or view the Pending Job Status Report from Option 8.7. The options are CNTL and PGMR.

JSREDIT

Indicates whether only control group members can edit a job stream member via Option 11. The options are:

YES

Specifies that any programmer can attempt to edit the member.

NO

(Default) Specifies that only control group members can edit the member.

LDISPLAY

Indicates whether a programmer is allowed to view all outstanding system link-edits and DB2 for z/OS and OS/390 bind requests from Options 10 and 8.6. The options are:

YES

Specifies that the programmer can view all requests.

NO

Specifies that the programmer can view only his own outstanding system link or DB2 for z/OS and OS/390 bind requests.

LIBRNAME

(CA Roscoe and CA Vollie.) Defines the name of the batch CA Librarian program as installed.

LINECOUNT

(TSO and VM/ESA.) Specifies the number of lines per page for report formatting. The value must be between 20 and 99 inclusive. The default is 59.

LINES

(CA Vollie only.) Specifies the number of lines per page for report formatting.

LKED

Defines how the production application system link-edit or DB2 for z/OS and OS/390 bind is performed when requested from Option 6 (LINK). See the Link-Edit Considerations section for your particular LIB/CCF system for more information on link editing. See the Initiating a Production Bind Request section in the *LIB/CCF User Guide* for more information on binds. The options are:

BATCH

Specifies that the programmer performs the link-edit or bind through the submission of a batch job.

ONLINE

(CA Roscoe and TSO) Specifies that the programmer performs the link-edit or bind interactively.

CNTL

Specifies that the control group performs the link or bind according to the CLINK value.

LNKNTEGRTY

(Except CA Vollie.) Indicates whether LIB/CCF checks for unresolved activity for this change request before allowing the link-edit or DB2 for z/OS and OS/390 bind request from Option 6. Unresolved activity includes outstanding module tracking records, module movement requests, job submission records, and duplicate system link requests for that change request.

YES

(Default) Does not allow the link-edit or bind request from Option 6 if there is any unresolved activity for the change request.

NO

Allows the link-edit or bind request from Option 6, regardless of any unresolved activity for the change request.

Note: If you specify NO, it is the user's responsibility to ensure that link or bind integrity is maintained

LOGLIMIT

The maximum number of logged out members a programmer can have. Specify a three-digit number from 001 to 999.

MDELETE

Indicates whether the control group can delete rejected login/logout requests. The options are:

YES

Indicates that the control group can delete rejected requests.

NO

Indicates that the control group cannot delete rejected requests.

MDISPLAY

Indicates whether a programmer is allowed to view all login/logout requests from Options 9 and 8.5. The options are:

YES

Indicates that the programmer can view all requests.

NO

Incidates that the programmer can view the status of only his own login/logout requests.

NOTIFY

(Except CA Vollie.) Indicates whether the programmer and programming manager are notified of opened, assigned, and closed change requests pertaining to them. The options are YES and NO. The SEND command is used in both CA Roscoe and TSO. The PUNCH command is used in VM/ESA

PMOVEMENT

Defines how members are moved to the production master file from the test master file when a programmer requests a login. If quality assurance (Q/A) libraries are defined in the library promotion path, this keyword also affects the movement of members into these libraries. The options are:

BATCH

Immediately submits a batch job that performs the copy, deletes the member from the test master, and performs optional subsequent processing. See the note below. Batch movement cannot be done in a VM/ESA batch machine.

ONLINE

Immediately copies the member and then deletes it from the test master. A batch job is then submitted to perform optional subsequent processing. See the note below.

CNTL

Submits a request to the control group. The CMOVEMENT keyword, described earlier in this section, determines whether the member is moved online or by a batch job.

Either all archive levels or the current level of the member are copied. In z/OS and OS/390 and VM/ESA, the SOURCE MOVE specification in the Library Chain Definition Function determines this. In LIB/CCF-AllFusion CA-Vollie, the COPY specification of the Master File Definition Table determines this.

At login time, LIB/CCF uses the language assigned to the member during its initial logout to determine what JCL skeleton (if any) performs subsequent processing. The LANG-to-JCL-skeleton relationship is defined in the Language Definition Table. For source members, subsequent processing can include either compiling from the production master file or copying the test object module to the production object library. In z/OS and OS/390 and VM/ESA, the OP SYS MOVE specification in the Library Chain Definition Function determines this.

In LIB/CCF-AllFusion CA-Vollie, the presence of a TEST RELOCATABLE LIBRARY in the Master File Definition Table determines this.

PRINT

(Except VM/ESA.) Defines how LIB/CCF print requests are serviced.

RPS

(CA Roscoe) Uses CA Roscoe Print Services. If RPS is not available, you must use BATCH.

VOLLIE

(All Fusion CA-Vollie) Uses the CA Vollie PRINT command. If PRINT support was not generated for CA Vollie, you must use BATCH.

ONLINE

(ISPF(TSO)) The print operation takes place immediately.

BATCH

Submits a batch job to print the report.

Note: The PRINT function is not applicable for ISPF(VM/ESA) because all VM/ESA users can choose their own method for printing on a separate panel in Option 0 (User Parameters).

Selecting PRINT on the User Parameters panel displays the Report Print Specification panel. You can make entries on this panel to direct where print requests are forwarded and to record other print information. Reports can be either spooled or directed to a user-defined VM/ESA file. This panel redisplays for modification, if necessary, whenever a print request is made.

PRTCLASS

(CA Vollie only.) Specifies the LST class used when reports are printed in batch.

SCHEDTRACE

(CA Roscoe and TSO.) Indicates whether a panel displaying the scheduling commands generated by LIB/CCF displays before performing the requested scheduling function. It is used for diagnostics.

ON

Indicates that the diagnostic panel displays.

OFF

(Default) Indicates that the diagnostic panel does not display.

SCHEDULER

(CA Roscoe and TSO.) Identifies CA 7 or CA Scheduler as available to schedule LIB/CCF jobs. Specify Unicenter CA-7 or CA Scheduler. There is no default. This keyword is optional. If omitted, jobs cannot be scheduled from LIB/CCF. See the “Scheduling z/OS and OS/390 Batch Jobs in LIB/CCF” chapter for details on scheduler use in LIB/CCF.

SYSOUT

(CA Roscoe and TSO.) Required for CA Roscoe. Specifies the SYSOUT class used when reports are printed in batch.

Optional for TSO. Specifies the SYSOUT class used when reports are printed in batch. If not specified, the default is class A.

TMOVEMENT

Defines how members are moved to the test master file from the production master file when a programmer requests a logout. If reject libraries (associated with quality assurance libraries) are defined in the library promotion path, this keyword also affects the movement of members into these libraries. The options are:

BATCH

Immediately submits a batch job that performs the copy. Batch movement cannot be done in a VM/ESA batch machine.

ONLINE

Immediately copies the member.

CNTL

Submits a request to the control group. The CMOVEMENT keyword, described earlier in this section, determines whether the member is moved online or by a batch job.

Note: Either all archive levels or the current level of the member are copied. In z/OS and OS/390 and VM/ESA, the SOURCE MOVE specification in the Library Chain Definition Function determines this. In LIB/CCF-AllFusion CA-Vollie, the COPY specification of the Master File Definition Table determines this.

UNIT

(CA Roscoe and TSO.) Required for CA Roscoe. Specifies the unit name used for allocation of temporary work data sets. For performance, the VIO unit name is recommended.

Optional for TSO. Specifies the unit name used for the allocation of temporary work data sets. If not specified, uses the default TSO unit name. For performance, the VIO unit name is recommended.

Note: Wide record master file users should be aware that the device, or the device being emulated when VIO, must support the LRECL of the largest members placed in a wide record master file.

UPPERCASE

(CA Vollie only.) Indicates the printed format of the VSE/ESA batch Change Request Status Report (CCFB101). This keyword is optional.

YES

Prints the report in upper case letters.

NO

(Default) Prints the report as is (with both upper and lower case letters).

System Profile Table Panel Commands

The commands available from this panel are:

=n (jump)

The value of *n* is an option number. Processes END before the jump (updates the table).

CANCEL

Ends processing and does not update the table.

END PF key

Ends processing and updates the table. For CA Vollie, you must have previously pressed Enter to save any table changes.

FIND string

Positions the display to the first occurrence of the specified string. Do not use delimiters.

NEXT

(CA Roscoe and CA Vollie) Positions the display to the next occurrence of the string that was previously specified by the FIND command.

RFIND

(TSO and VM/ESA.) Positions the display to the next occurrence of the string that was previously specified by the FIND command.

SORT

(Except CA Vollie.) Sorts the table by keyword. (The table is automatically sorted when saved, however, the command can be useful when adding new keywords.)

User Definition Table (Option 12.1)

The User Definition Table defines to LIB/CCF the users who can open change requests from Option 1 (OPEN) and the applications for which they can open requests.

User Definition Table Panel Fields

The panel fields are:

USER KEY

(CA Roscoe and CA Vollie.) Defines the user's LIB/CCF key, either the first eight characters of the user's CA Roscoe key or the LIB/CCF key on the CA Roscoe key Conversion Table (Option 12.8).

You can also specify a group ID, as defined in Option 12.12 (GROUPS). If specified, the group ID must be defined in Option 12.12 before you can use it in this table.

For CA Vollie, use either the first eight characters of the user's CICS operator name or the LIB/CCF key from the OPNAME Conversion Table (Option 12.8).

USER ID

(TSO and VM/ESA.) Defines the user's LIB/CCF ID, either the TSO ID or the LIB/CCF key defined in the SYSUID Conversion Table (Option 12.8). You can also specify a group ID, as defined in Option 12.12 (GROUPS). If specified, the group ID must be defined in Option 12.12 before you can use it in this table.

AUTHORIZATION CODE

(Optional.) If an authorization code is specified, the user must enter this code when opening a request for the specified application system.

AUTHORIZED APPLICATION

Specifies the name of the application system for which the user can open a request. There should be an entry for each application system for which the user is authorized.

ALL

The user can open a request for any application system. Specifying ALL requires only one entry in the table.

User Definition Table Panel Commands

The commands available from this panel are:

=n (jump)

The value of *n* is an option number. Processes end before the jump (updates the table).

CANCEL

Ends processing and does not update the table.

END PF key

Ends processing and updates the table. For CA Vollie, you must have previously pressed Enter to save any table changes.

FIND string

Positions the display to the first occurrence of the specified string. Do not use delimiters.

NEXT

(CA Roscoe and CA Vollie.) Positions the display to the next occurrence of the string that was previously specified by the FIND command.

RFIND

(TSO and VM/ESA.) Positions the display to the next occurrence of the string that was previously specified by the FIND command.

SORT [USER|APPL]

(Except CA Vollie.) Sorts the table by either the USER or AUTHORIZED APPLICATION. The table is automatically sorted when saved. Specified with no operand, the table sorts by USER.

Manager Definition Table (Option 12.2)

The Manager Definition Table defines to LIB/CCF the programming managers and the application systems for which they are responsible.

Manager Definition Table Panel Fields

The panel fields are:

MGRS ID

Defines the application manager's LIB/CCF key. This is one of the following:

- For CA Roscoe—The first eight characters of the CA Roscoe key or the LIB/CCF key defined in the CA Roscoe Key Conversion Table (Option 12.8).
- For CA Vollie—The first eight characters of the manager's CICS operator name, or the LIB/CCF key defined in the OPNAME Conversion Table (Option 12.8).
- For ISPF(TSO) or ISPF(VM/ESA)—The TSO or VM/ESA user ID or the LIB/CCF ID defined in the SYSUID Conversion Table (Option 12.8).

You can also specify a group ID, as defined in Option 12.12 (GROUPS). If specified, the group ID must be defined in Option 12.12 before you can use it in this table.

AUTHORIZATION CODE

(Optional.) If a code is specified, the application manager must enter this code when assigning a change request to a programmer.

APPLICATION

Indicates the name of the application system for which the manager is responsible. An application can be listed only once in the table.

CHAIN GROUP

Indicates the name of the library promotion chains used when change requests are opened for this application. Different applications can share the same chain group.

- For CA Vollie, the chain group is required and corresponds to the chain group specified in the Master File Definition Table (Option 12.4).
- For CA Roscoe, ISPF(VM/ESA), and ISPF(TSO), the chain group is required and corresponds with the chain group specified in LCDF (Option 12.11).

Note: If nothing is entered in this field, LIB/CCF automatically sets the field to blanks, which is considered a valid chain group name. Any application with a blank chain group uses the chains defined in LCDF that also have a blank chain group.

Manager Definition Table Panel Commands

The commands available from this panel are:

=n (jump)

The value of *n* is an option number. Processes end before the jump (updates the table).

CANCEL

Ends processing and does not update the table.

END PF key

Ends processing and updates the table.

For CA Vollie, you must have previously pressed Enter to save any table changes.

FIND string

Positions the display to the first occurrence of the specified string. Do not use delimiters.

NEXT

(CA Roscoe) Positions the display to the next occurrence of the string that was previously specified by the CA Vollie FIND command.

RFIND

(TSO and VM/ESA.) Positions the display to the next occurrence of the string that was previously specified by the FIND command.

SORT [MGR|APPL|CHAIN]

(Except CA Vollie.) Sorts the table by either the MGR ID, APPLICATION, or CHAIN GROUP. The table is automatically sorted when saved. Specified with no operand, the table sorts by MGR.

Programmer Definition Table (Option 12.3)

The Programmer Definition Table defines to LIB/CCF the programmers and the application managers to whom they report.

Programmer Definition Table Panel Fields

The panel fields are:

MGRS ID

Defines the manager's LIB/CCF key, one of the following:

- For CA Roscoe—The first eight characters of the CA Roscoe key or the LIB/CCF key defined in the CA Roscoe Key Conversion Table (Option 12.8).
- For CA Vollie—The first eight characters of the manager's CICS operator name or the LIB/CCF key defined in the OPNAME conversion table (Option 12.8).
- For ISPF(TSO) or ISPF(VM/ESA)—The TSO or VM/ESA user ID or the LIB/CCF ID defined in the SYSUID Conversion Table (Option 12.8).

You can also specify a group ID, as defined in Option 12.12 (GROUPS). If specified, the group ID must be defined in Option 12.12 before you can use it in this table.

PGMR ID

Defines the programmer's LIB/CCF key, one of the following:

- For CA Roscoe—The first eight characters of the CA Roscoe key, or the LIB/CCF key defined in the CA Roscoe Key Conversion Table (Option 12.8).
- For CA Vollie—The first eight characters of the programmer's CICS operator name or the LIB/CCF key defined in the OPNAME Conversion Table (Option 12.8).
- For ISPF(TSO) or ISPF(VM/ESA)—The TSO or VM/ESA user ID, or the LIB/CCF ID defined in the SYSUID Conversion Table (Option 12.8).

You can also specify a group ID, as defined in Option 12.12 (GROUPS). If specified, the group ID must be defined in Option 12.12 before you can use it in this table.

Programmer Definition Table Panel Commands

The commands available from this panel are:

=n (jump)

The value of *n* is an option number. Processes end before the jump (updates the table).

CANCEL

Ends processing and does not update the table.

END pf key

Ends processing and updates the table. For CA Vollie, you must have previously pressed Enter to save any tables changes.

FIND string

Positions the display to the first occurrence of the specified string. Do not use delimiters.

NEXT

(CA Roscoe and CA Vollie.) Positions the display to the next occurrence of the string that was previously specified by the FIND command.

RFIND

(TSO and VM/ESA.) Positions the display to the next occurrence of the string that was previously specified by the FIND command.

SORT [MGR|PGMR]

(Except CA Vollie.) Sorts the table by either the MGR ID or PGMR ID. The table is automatically sorted when saved. Specified with no operand, the table sorts by MGR ID.

Master File Definition Table (Option 12.4: AllFusion CA-Vollie Only)

The Master File Definition Table defines the production master files from which members can be logged out. For each pair of master files, production and test, there is one entry on this table with a unique entry number. You can add or delete entries while viewing this panel.

Important! All master files named in this table must be defined in the CICS FCT. Refer to the CA Vollie Installation Procedures Guide for sample FCT entries for CA Librarian master files.

Master File Definition Table Panel Fields

The panel fields are:

ENTRY

Displays the entry number for the currently displayed entry. You can overtype the entry number you want to display. To add a new entry, blank this field and complete the rest of the panel. To delete the entry currently displayed, type **D** in this field.

ENTRY TYPE

Displays the type of entry currently displayed. The entry types are P, Q, F, and R:

P

Relates production libraries to test libraries.

Q

Relates one level of a QA library to the next highest level of a QA library, a production library to the highest or final QA library, and a QA library to a reject library.

F

Relates the first QA library in a chain to the eventual production library.

R

Relates QA libraries to reject libraries.

To create a different type than is currently displayed, enter the new type and press Enter. The appropriate headings for that entry type display.

Note: For details on QA processing with LIB/CCF-CA Vollie. see the Implementing Quality Assurance Procedures chapter.

TOTAL ENTRIES

(Display only) Total number of entries in the table. The administrator cannot modify this field.

CHAIN GROUP

Defines a single library chain or group of library chains used when change requests are opened for the related applications defined in Option 12.2. The chain group cannot contain the slash (/) character.

PRODUCTION MASTER FILE-ID

The file ID of the production CA Librarian master file that LIB/CCF is to control.

STATUS

CA Librarian MCD security status that is assigned to the member when added to this master file. You can specify this field ONLY if the master file has an MCD assigned to it.

COPY

The method used to copy the member to this master file. The COPY specification is relevant only when the movement operation takes place in the BATCH environment. Online moves are always done using IMPORT/UPDATE of the current level of the member. The options are:

U

Uses the batch CA Librarian UTILITY COPY function. UTILITY COPY copies all archive levels, replacing the entire module on the receiving master file.

S

Uses LIBDCOPY. LIBDCOPY copies only the most current archive level of the module and updates an existing module or add a new module.

ACCESS

Defines the type of access allowed from CA Vollie when the CA Vollie CCF security facility is implemented.

A

Any access allowed.

B (Browse)

DISPLAY and LMI only, without LMI update capabilities

R

(Default) Read, Browse, and Copy. DISPLAY, COPY, IMPORT, and LMI only, without LMI update capabilities

N

None.

TEST MASTER FILE-ID

The file ID of the related test master file to which members are logged out. This field appears only for P type entries.

LOCK

The CA Librarian MCD security status that is assigned to the member on the test master file when the login function is performed. This cannot be specified unless the test master file has an MCD code assigned to it. It applies only to P type entries, and then only when no QA libraries are used.

PRODUCTION RELOCATABLE LIBRARY

(Optional) The production relocatable library to be the target of production compiles or to which the test object member is copied if a test relocatable library is specified. Specify the library.sublibrary name for VSE/ESA.

TEST RELOCATABLE LIBRARY

(Optional) Defines the test relocatable library and indicates that the object is copied from this library to the production relocatable library rather than compiling the member after it is moved to the production master. This appears only for P type entries. Specify the library.sublibrary name for VSE/ESA.

Computer Associates does not recommend the use of a test object. If a test object is copied to the production object and then linked, the Source to Load Audit Trail (SLAT) variables do not match when LIBDSCAN checks the source master file against the core image library. See the Source to Load Member Verification section in the "LIB/CCF Model System" chapter for details on the SLAT variables.

TO Q/A MASTER FILE-ID

Appears for Q and F type entries. The file ID of the Quality Assurance (QA) master file where members residing in the From QA Master File are moved.

For F entries, the name of the first QA library in a Test to Production library chain.

FROM Q/A MASTER FILE-ID

For Q entries, the file ID of the Quality Assurance library from a previous level of QA. Members residing in this library are processed to the next higher level as defined in the TO Q/A entry.

For R entries, the file ID of the QA library to relate to the reject library specified in the entry.

OP SYS MOVE

The action to take to produce a production object module.

C

(Default). Compile the member after moving it to this library, using the JCL skeleton associated with the language that was assigned to the member when it was initially logged out. (The language-to-JCL-skeleton relationship is defined in Option 12.5.) This library must have an associated object library or an error results during login.

Y

Copy the object module from the previous library. This library and the previous library must both have an associated object library or an error results during login.

N

No action takes place after the member is moved.

TO REJECT MASTER FILE-ID

Identifies the reject library where members are moved when rejected from the related Q/A master file.

FROM PRODUCTION MASTER FILE-ID

Identifies the production master file where the TO Q/A library is the first QA library in a chain.

Master File Definition Table Panel Commands

The commands available from this panel are:

=n (jump)

The value of *n* is an option number. Updates the table before the jump.

CANCEL

Ends processing and **does not** update the table.

PF3/PF15 key

Ends processing and updates the table. You must have previously pressed Enter to save any table changes.

FIND string

Positions the display to the first occurrence of the specified string. Do not use delimiters.

NEXT

Positions the display to the next occurrence of the string that was previously specified by the FIND command.

Language Definition Table (Option 12.5)

The Language Definition Table correlates JCL skeletons with a LANGUAGE. When a member is logged out for the first time through LIB/CCF, the programmer assigns a language to it through the History Create panel. At login, LIB/CCF uses this language to determine which JCL skeleton is submitted to compile or otherwise process the member.

Language Definition Table Panel Fields

The panel fields are:

DESCRIPTION

(Optional) A description of what the skeleton is used for. Can be up to 30 characters and contain blanks. This description appears on the HELP panel available from the Option 4 and 5 History Create/Modify panel when an invalid language is entered.

LANGUAGE

Any one- to eight-character language name. When a member is logged out for the first time, the History Create/Modify panel prompts the programmer for member information, including a LANGUAGE code. You must supply one of the languages from this table. Once the History Create panel is filled in and the programmer presses Enter, a history member is added to the history master file defined for the production master file. The LANGUAGE is stored as a data record in the history member and determines which JCL skeleton is used to compile the member when it is logged in.

Note: Do not confuse this language name with the three-character language that can optionally be assigned to a member (with the CA Librarian -LANG control statement) and stored internally with a member's control information.

PRTY

(Except CA Vollie.) Any number from 0 to 999. This number orders members in the Group Processing selection list, available from Options 5 and 9. (000 is the highest priority, 999 is the lowest priority. Members given a 000 priority are processed before other members.) This can be useful when the order in which members are processed is important.

SKELETON

Either the name of the skeleton JCL member for the compile or an entry of NOP to indicate that a compile is not performed.

- For CA Roscoe—The member must exist in the library where LIB/CCF was installed.
- For ISPF(TSO) and ISPF(VM/ESA)—The skeleton must exist in the skeleton library defined in the ISPSLIB concatenation.
- For CA Vollie— The skeleton must exist as a member of the VOICE library where CCF was installed.
- For ISPF(VM/ESA)—The SKELETONS keyword has subkeywords of z/OS and OS/390, VM/ESA, and VSE/ESA. Specify the name of the skeleton JCL member for the operating systems where the job is submitted. If an operating system is left blank, the associated language cannot be processed in that operating system.

TESTSKEL

The name of the test skeleton JCL member to execute when the Option 5 (login) TEST command is issued. The JCL skeleton can perform any processing before member login (for example, a test compile). You can supply your own JCL skeleton or use one of the sample skeletons. Sample skeletons are provided for Assembler, COBOL, and PL1 to compile the test member from the test master file. The sample members are:

- For ISPF:
  - $CCFJT10
  - $CCFJT11
  - $CCFJT12
- For CA Roscoe:
  - $CHGJT10
  - $CHGJT11
  - $CHGJT12
- For CA Vollie:
  - $CCVJT10
  - $CCVJT11
  - $CCVJT12

You can specify the same skeleton for both the SKELETON and TESTSKEL panel fields.

The test JCL skeleton must use only the extension variables documented in the “Extension Variables” appendix.

Specify an entry of NOP to indicate that a compile is **not** performed.

For ISPF(VM/ESA), the TESTSKEL keyword has subkeywords of z/OS and OS/390, VM/ESA, and VSE/ESA. Specify the name of the skeleton JCL member for the operating systems where the job is submitted. If an operating system is left blank, the associated language cannot be processed in that operating system.

Language Definition Table Panel Commands

The commands available from this panel are:

=n (jump)

The value of *n* is an option number. Processes end before the jump (updates the table).

CANCEL

Ends processing and *does not* update the table.

END pf key

Ends processing and updates the table.

For CA Vollie, you must have previously pressed Enter to save any table changes.

FIND string

Positions the display to the first occurrence of the specified string. Do not use delimiters.

NEXT

(CA Roscoe and CA Vollie.) Positions the display to the next occurrence of the string that was previously specified by the FIND command.

RFIND

(TSO and VM/ESA.) Positions the display to the next occurrence of the string that was previously specified by the FIND command.

SORT [LANG|TEST|SKEL|PRTY|DESC]

(Except CA Vollie.) Sorts the table by either the LANG, SKEL, TEST, PRTY, or DESC. The table is automatically sorted when saved. Specified with no operand, the table sorts by LANG.

Production Master File Definition Table (Option 12.6: AllFusion CA-Vollie Only)

The Production Master File Definition Table defines the core image library that is associated with each production master.

Production Master File Definition Table Panel Fields

The panel fields are:

PRODUCTION MASTER FILE-ID

The file ID of the production CA Librarian master.

PRODUCTION CORE IMAGE LIBRARY

The name of the production core image library where link-edited members from the production master are placed. Specify the sublibrary name for VSE/ESA.

PRODUCTION RELOCATABLE LIBRARY

The production relocatable library from which the link-edited members are taken. Specify the library.sublibrary for VSE/ESA.

PRODUCTION BACKUP CORE IMAGE LIBRARY

(Optional) The file name for the backup core image library where the production phase is copied before the link edit of a new production phase. The copy is done only if this field is supplied. Specify the file name for VSE/ESA SP1 or the library.sublibrary name for VSE/ESA SP2.

Production Master File Table Panel Commands

The commands available from this panel are as follows:

=n (jump)

The value of *n* is an option number. Processes end before the jump (updates the table).

CANCEL

Ends processing and *does not* update the table.

PF3/PF15 key

Ends processing and updates the table. You must have previously pressed Enter to save any table changes.

FIND string

Positions the display to the first occurrence of the specified string. Do not use delimiters.

NEXT

Positions the display to the next occurrence of the string that was previously specified by the FIND command.

History Master File Definition Table (Option 12.7: AllFusion CA-Vollie Only)

The History Master File Definition Table defines the associated history master for each production master.

History Master File Definition Table Panel Fields

The panel fields are:

PRODUCTION MASTER FILE-ID

The file ID of the production CA Librarian master file.

HISTORY MASTER FILE-ID

The file ID of the related history master file.

STATUS

CA Librarian MCD security status that is assigned to members on the history master file. You can specify this field only if the History Master has an MCD code assigned to it.

History Master File Definition Table Panel Commands

The commands available from this panel are:

=n (jump)

The value of *n* is an option number. Processes end before the jump (updates the table).

CANCEL

Ends processing and *does not* update the table.

PF3/PF15 key

Ends processing and updates the table. You must have previously pressed Enter to save any table changes.

FIND string

Positions the display to the first occurrence of the specified string. Do not use delimiters.

NEXT

Positions the display to the next occurrence of the string that was previously specified by the FIND command.

ID Conversion Table (Option 12.8)

The ID Conversion table lets LIB/CCF convert an CA Roscoe key, TSO SYSUID, VM/ESA SYSUID, or AllFusion CA-Vollie CICS OPNAME to a LIB/CCF ID. Use ID conversion if your site requires any of the following:

- To use real names as LIB/CCF IDs when user IDs are not meaningful. For example, real names can be more meaningful in the LIB/CCF reports to someone unfamiliar with your site's user IDs.
- To define more than one manager per application (since Option 12.2 (MANAGER) allows only one manager to be defined per application). For example, if the PAYROLL application is managed by two people, JONES and ROBERTS, convert both IDs to PAYMGR in this table. In Option 12.2, define PAYMGR as the manager for the PAYROLL application. Any change requests opened under the PAYROLL application are placed in the Option 2 (ASSIGN) queue for *both* JONES and ROBERTS. In this way, either manager can assign the change request, thereby removing it from both manager's queue.
- For CA Vollie, when the first eight characters of the user's CICS operator name are not unique. For example, if the operator naming convention is SYSTEMS.lastname, then the first eight characters are not unique to each user. In this case, entries are required in the table to define a unique LIB/CCF ID for each user.
- For CA Roscoe, when the first eight characters of the CA Roscoe key are not unique. For example, if the CA Roscoe ID naming convention is SYSTEMS.lastname, then the first eight characters are not unique to each user. In this case, entries are required in the table to define a unique LIB/CCF ID for each user.

If a user ID is defined in this table, the conversion is performed automatically when the user signs on to LIB/CCF. It is in effect for all LIB/CCF processing. To demonstrate, let us convert an CA Roscoe key of SYSPG01 to a LIB/CCF key of SMITH. When SYSPG01 signs on to LIB/CCF, that user is known to LIB/CCF only as SMITH. Therefore, SMITH must be defined in the appropriate LIB/CCF tables as a user, manager, programmer, control key, or emergency key.

Important! The one exception to this is for the ISPF (VM/ESA and TSO) CNTLID (control ID), EMERID (emergency ID), and ADMINID (administrator ID) keywords in Option 12.0, which must specify the real SYSUID, not the converted CCF ID.

Note: You cannot convert group IDs, as defined in Option 12.12 (GROUPS).

CA Roscoe Key Conversion Table

Entries in this table associate an CA Roscoe key with a LIB/CCF key.

ID Conversion Table Panel Fields

The panel fields are:

ROSCOE KEY

The CA Roscoe key for the user.

CCF KEY

The one- to eight-character LIB/CCF key for the user. It must be different from all other LIB/CCF keys. If an CA Roscoe user does not have an entry in this table, the first eight characters of that user's CA Roscoe key are used as the LIB/CCF key.

ID Conversion Table Panel Commands

The commands available from this panel are:

=n (jump)

The value of *n* is an option number. Processes end before the jump (updates the table).

CANCEL

Ends processing and *does not* update the table.

END pf key

Ends processing and updates the table.

FIND string

Positions the display to the first occurrence of the specified string. Do not use delimiters.

NEXT

Positions the display to the next occurrence of the string that was previously specified by the FIND command.

SORT [ROSCOE|CCF]

Sorts the table by either the ROSCOE or CCF ID. The table is automatically sorted when saved. Specified with no operand, the ID stores the table.

ISPF SYSUID Conversion Table

Entries in this table associate a LIB/CCF key with a TSO or VM/ESA SYSUID.

ISPF SYSUID Conversion Table Panel Fields

The panel fields are:

TSO OR VM/ESA ID

The user ID for this user.

CCF ID

A unique one- to eight-character LIB/CCF ID.

If there is no entry in this table for a particular user ID, the user ID itself is used as the CCF ID.

ISPF SYSUID Conversion Table Panel Commands

The commands available from this panel are:

=n (jump)

The value of *n* is an option number. Processes end before the jump (updates the table).

CANCEL

Ends processing and *does not* update the table.

END pf key

Ends processing and updates the table.

FIND string

Positions the display to the first occurrence of the specified string. Do not use delimiters.

RFIND

Positions the display to the next occurrence of the string that was previously specified by the FIND command.

SORT [TSO|CMS|CCF]

Sorts the table by either the TSO, VM/ESA or CCF ID. The table is automatically sorted when saved. Specified with no operand, the table is sorted by the TSO or VM/ESA ID, depending on which system is used.

CA Vollie OPNAME Conversion Table

Entries in this table associate a CICS operator name with a LIB/CCF key.

OPNAME Conversion Table Panel Fields

The panel fields are:

OPERATOR NAME

The CICS operator name for the user.

CCF KEY

The one- to eight-character LIB/CCF key for the user.

If an CA Vollie user does not have an entry in this table, the first eight characters of that user's operator name are used as the LIB/CCF key.

OPNAME Conversion Table Panel Commands

The commands available from this panel are:

=n (jump)

The value of *n* is an option number. Processes end before the jump (updates the table).

CANCEL

Ends processing and *does not* update the table.

PF3/PF15 key

Ends processing and updates the table. You must have previously pressed Enter to save any table changes.

FIND string

Positions the display to the first occurrence of the specified string. Do not use delimiters.

NEXT

Positions the display to the next occurrence of the string that was previously specified by the FIND command.

Change Request Reassignment (Option 12.9)

Change Request Reassignment lets the System Administrator reassign change requests to different programmers or managers. Option 2 (ASSIGN) allows a manager to use the REASSIGN command to reassign change requests from one programmer to another, provided that both programmers are defined as reporting to him in Option 12.3 (PROGRAMMER).

Change Request Reassignment Panel Fields

The panel fields are:

REQUEST ID

The change request number. Leave this field blank to reassign all the change requests that were assigned to the specified programmer or manager.

PROGRAMMER

The CCF ID of the programmer to whom the change request is currently assigned. You can specify a group ID from option 12.12 (except for CA Vollie sites).

REASSIGN TO PROGRAMMER

The CCF ID of the programmer to whom the requests are reassigned. You can specify a group ID from option 12.12 (except for CA Vollie sites).

MANAGER

The CCF ID of the manager to whom the Change Request is currently assigned. You can specify a group ID from option 12.12 (except for AllFusion CA-Vollie sites).

REASSIGN TO MANAGER

The CCF ID of the manager to whom the requests are reassigned. You can specify a group ID from option 12.12 (except for CA Vollie sites).

Change Request Reassignment Panel Commands

The commands available from this panel are:

=n (jump)

The value of *n* is an option number. Processes end before the jump (updates the table).

END pf key

Ends processing.

Master File Information Table (Option 12.10: AllFusion CA-Vollie Only)

This list contains information CA Vollie requires to access CA Librarian master files online. It also contains information needed to build proper JCL for batch operations.

Every CA Librarian master file defined in any other table must have one (and only one) entry in this table.

Master File Information Table Panel Fields

The panel fields are:

ENTRY

Displays the entry number for the currently displayed entry. You can overtype the entry number you want to display. To add a new entry, blank this field and complete the rest of the panel. To delete the currently displayed entry, enter **D** in this field.

TOTAL ENTRIES

Total number of entries in the table. This field is for display only. The administrator cannot modify it.

LIBRARIAN MASTER FILE ID

The file ID of the CA Librarian master file.

FCT NAME

The name used in the CICS FCT entry for this master file.

FILE ORGANIZATION

The type of master: BDAM or VSAM.

BASE

The base CA Librarian management code for this CA Librarian master file. (See the *Security Administration Guide* for a description of management codes and their restrictions.)

VOLSER

The volume serial number where this master resides. Although it relates only to BDAM masters, it is also required for VSAM masters to satisfy the JCL requirements in the JCL skeletons.

VSAM CATNAME

The name of the VSAM catalog containing the entry for this master file. (For VSAM masters only.)

Master File Information Table Panel Commands

The commands available from this panel are:

=n (jump)

The value of *n* is an option number. Processes end before the jump (updates the table).

CANCEL

Ends processing and *does not* update the table.

PF3/PF15 key

Ends processing and updates the table. You must have previously pressed Enter to save any table changes.

FIND string

Positions the display to the first occurrence of the specified string. Do not use delimiters.

Library Chain Definition (Option 12.11: Except AllFusion CA-Vollie)

The Library Chain Definition Function (LCDF) defines production, Q/A, reject, and test master files and their associated libraries (object, load, and history) as a library chain. A library chain is a group of logically related libraries. The relationships contain information about the type of each library (production, test, reject, or Q/A), how to move modules between these libraries, and other information. LCDF lets you arrange library chains in chain groups and relate them to an application system through a chain group identifier. Each master file is specified in its logical order in the promotion hierarchy of the library chain and is designated as Production, Quality Assurance, Reject, or Test. Each chain has a unique identification number and is a subset of a chain group.

LCDF is available for the CA Roscoe, ISPF(TSO), and ISPF(VM/ESA) environments. Similar functions are performed for the CA Vollie environment by options 12.4, 12.6, 12.7, and 12.10.

LCDF Selection List Processing

LCDF supports selection list processing for certain panel fields and functions. Each field or function described in the following sections indicates how to obtain a selection list, but generally, use an asterisk or blank for a complete list of libraries or chains. There are two additional ways of using wildcard specification.

LIBR*

Place an asterisk at the end of the specification to select all library names starting with the characters preceding the asterisk.

L*BR

Place an asterisk in a character position to select, in this example, all libraries with four character names, an L in the first position, B in the third position, and R in the fourth position.

You can use these two types together. For example, the specification of L**B* selects libraries that have an L as the first character, anything as the second and third characters, a B as the fourth character, and any number of unspecified characters after that.

LCDF Panel Commands

You can use the following commands in the various Library Chain Definition Functions:

BACK

Redisplays the previous panel in the function, if possible.

CANCEL

Cancels the function without an update.

END

On the first panel of each function, CANCELs without an update. On functions where indicated, proceeds to the next panel in sequence. In all other cases, FINISHes with an update.

FINISH

Skips all remaining panels that are normally displayed for the function, if possible, and completes with an update.

Select Option 12.11 from the LIB/CCF Administrator menu. The LCDF menu appears.

```
                LIBRARY CHAIN DEFINITION

OPTION===>

 1 CREATE CHAIN   - Create library chain
 2 MODIFY CHAIN   - Modify library chain
 3 MODIFY LIBRARY - Modify library definition
 4 DEACTIVATE     - Deactivate library chain
 5 DELETE         - Delete deactivated library or chain
```

The following sections describe the Option 12.11 functions. See the Glossary for the meaning of any unfamiliar terms.

Note: Do not alter chains until all related outstanding members are processed to completion, or the logout has been deleted. Changing chains when members are outstanding can cause unpredictable results.

Create a Library Chain

The CREATE function defines a new library chain to CCF. You can model the chain after an existing chain. When you select Option 12.11.1 (Create library chain), a series of panels appears. When you fill in all the necessary information, LCDF creates the library chain.

```
                        CREATE CHAIN
COMMAND===>

                      CHAIN NUMBER: 00001

   CHAIN GROUP===> PAYROLL

   DESCRIPTION LIBRARY PROMOTION PATH FOR PAYROLL

CHAIN MODEL===> (asterisk for selection list)
```

CHAIN NUMBER

LCDF generates the new chain number automatically for the chain being created.

CHAIN GROUP

Corresponds to the chain group defined in Option 12.2 for one or more applications. If you leave this field blank, LIB/CCF automatically sets the field to blanks, which is considered a valid chain group name. See the Manager Definition Table (Option 12.2) section in the “Administrator Functions in LIB/CCF” chapter for details on the CHAIN GROUP keyword of Option 12.2.

DESCRIPTION

(Required.) A brief description of the chain.

CHAIN MODEL

(Optional.) To specify an existing chain to use as a model, enter its chain number in the Chain Model field. Alternatively, enter ***** to obtain a selection list of all chains. Use an *S* to select the chain to use as a model. LCDF creates a copy that is used for building the new chain.

Chain Description Panel

The chain description panel allows modification of the chain group and chain description.

```
                    CHAIN DESCRIPTION

COMMAND===>

                    CHAIN NUMBER: 00001

   CHAIN GROUP===> PAYROLL

   DESCRIPTION===> LIBRARY CHAIN FOR PAYROLL APPLICATIONS
```

Chain Structure Panel

The chain structure panel graphically defines the libraries in the chain. Each chain must have at least a PROD and TEST type library.

```
                      CHAIN STRUCTURE ROW 1 OF 6

                     "END" TO CONTINUE

COMMAND ===>                                  SCROLL ===> PAGE

               CHAIN NUMBER: 00001        CHAIN GROUP: PAYROLL

   DESCRIPTION: LIBRARY CHAIN FOR PAYROLL APPLICATIONS

ACTION  TYPE        LIBRARY
        PROD        PAYROLL-PROD
        Q/A         PAYROLL-STAGING
        REJ         PAYROLL-REJECT
        Q/A         PAYROLL-QA-TEST
        REJ         PAYROLL-TEST
        TEST        PAYROLL-TEST
A
```

ACTION

(Optional) For inserting or deleting libraries. One of the following:

Inn

Insert a number (*nn*) of lines. If you do not specify a number, one line is inserted.

Dnn

Delete a number (*nn*) of lines. If you do not specify a number, one line is deleted.

DD

Delete a block of libraries, starting with the line where the first DD is found and ending with the line where the next DD is found.

TYPE

Library type, one of the following:

PROD

The production library (required).

Q/A

Intermediate quality assurance libraries (optional).

REJ

Reject libraries, where the module goes if rejected from the corresponding Q/A library (optional).

TEST

The test library (required).

LIBRARY

(Required) The 1- to 44-byte library name (not necessarily the data set name). Alternatively, enter an ***** or wildcard specification to obtain a selection list of all the libraries of the type specified. For example, if you enter ***** for the library type of PROD, a list of all libraries previously defined as PROD displays. Enter *S* to select a library from the list. Press Enter to return to the Chain Structure panel with the selected library name entered in the appropriate LIBRARY field.

Note: PROD and Q/A types can refer to the same library. REJ and TEST types can refer to the same library. However, you cannot define a PROD or Q/A type library as a REJ or TEST type library in the same chain or any other chain. Similarly, you cannot define a REJ or TEST type library as a PROD or Q/A library in the same chain or any other chain.

Chain Attributes Panel

The chain attributes panel lets you specify how members are copied, their status, the action taken with operating system files, and the promotion path for rejected members.

```
                    CHAIN ATTRIBUTES                        ROW 1 OF 6

                  "END" TO CONTINUE
COMMAND ===>                                          SCROLL ===> PAGE
            CHAIN NUMBER: 00001    CHAIN GROUP: PAYROLL
  DESCRIPTION: LIBRARY CHAIN FOR PAYROLL APPLICATIONS

LIBRARY
                       SOURCE   STATUS   OP SYS  PROMOTE  LOCK   LOGIN
                        MOVE             MOVE                    MOVE
PAYROLL-PROD
                         C        1        C       *       *      *
  PAYROLL-STAGING
                         A        T        N       *       *      *
    PAYROLL-REJECT
                         A        T        N       F       T      Y
  PAYROLL-QA-TEST
                         A        T        N       *       *      *
    PAYROLL-TEST
                         A        T        N       F       T      Y
PAYROLL-TEST
                         A        T        *       *       T      Y
```

LIBRARY

(Display only) The library name for the CA Librarian master file at this location in the chain. PROD and TEST libraries display at the left margin. Q/A libraries are indented two spaces. REJ libraries are indented four spaces.

SOURCE MOVE

(Required) Specifies whether the CA Librarian member is moved into this library with all its archived levels or only the current level.

C

Move only the current archived level of the member. If the member exists on the receiving file, it is updated. If not, it is added. In z/OS and OS/390, LIBRCOPY is executed. In VM/ESA, a service routine is executed. In VSE/ESA, LIBDCOPY is executed.

A

Move all archive levels of the member. If the member exists on the receiving file, it is replaced. In z/OS and OS/390 or VSE/ESA, CA Librarian Utility Copy is executed.

In addition to this field, three Option 12.0 (PROFILE) keywords affect the method in which members are moved into the library (that is, by a batch job or online, by the programmer, or the control group). These keywords are CMOVEMENT, PMOVEMENT, and TMOVEMENT. See the System Profile Table section in the "Administrator Functions in LIB/CCF" chapter for details.

Note: In the LIB/CCF-ISPF(VM/ESA) environment, you must specify **C** if this library is a VM/ESA formatted master file (that is, initialized with the LIB/CMS LIBMAST command) or if movement into this library is performed online (as specified by the Option 12.0 keyword MOVEMENT).

STATUS

The MCD security status assigned to the member when it is moved to this library. Can be T, 0, 1, or 2, corresponding to TEST, PROD0, PROD1, or PROD2. When left blank, a default of T, or TEST, is assumed.

OP SYS MOVE

(Required) How members of operating system files are moved at login.

C

Compile the member after moving it to this library, using the JCL skeleton associated with the language that was assigned to the member when it was initially logged out. The language-to-JCL-skeleton relationship is defined in Option 12.5. This library must have an associated object library or an LCDF error results during login.

L

Copy the load module from the previous library. This library and the previous library must both have an associated load library or an LCDF error results during login.

O

Copy the object module from the previous library. This library and the previous library must both have an associated object library, or an LCDF error results during login.

N

No action takes place after the member is moved.

*** (asterisk)**

This attribute does not apply to the type of library specified.

PROMOTE

(Required for Reject libraries.) Where a module promoted out of a REJ library is moved.

F

Move the rejected member from the reject library back to the first Q/A library in the chain.

B

Move the rejected member from the reject library directly back to the Q/A library it was rejected from.

*** (asterisk)**

This attribute does not apply to the type of library specified.

LOCK

(Required for Test and Reject libraries.) The MCD security status assigned to the member when locking it in this library after a login was requested of the control group. Allowable values are T, 0, 1, and 2, corresponding to the MCD security status codes of TEST, PROD0, PROD1, and PROD2. When left blank, a default of T, or TEST, is assumed.

The asterisk (*) does not apply to the type of library specified.

LOGIN MOVE

Applicable only if:

- A control group is defined, *and*
- A Q/A library is defined in the chain.

Controls movement into the Q/A library. Required for test libraries (even if no Q/A library is defined in the chain) and reject libraries.

Y

The login from Option 5 moves the member into the Q/A library from the test or reject library.

N

The login from Option 5 does nothing other than create a request for the control group. It does not move the member into the Q/A library from the test or reject library.

* (a**sterisk)**

This attribute does not apply to the type of library specified.

Associated Files Panel

The associated files panel lets you define the related object, load, and DBRM libraries for each operating system environment. All files are optional. The file name need not be the real data set name.

A production library must have an associated object and load library if it is presented to the programmer as an available file combination for the System Link Request (Option 6). Similarly, a production library must have an associated object, DBRM, and load library if it is presented to the programmer as an available file combination for the System Bind Request (Option 6).

```
                    ASSOCIATED FILES

COMMAND ===>
              CHAIN NUMBER: 00001 CHAIN GROUP: PAYROLL
  DESCRIPTION: LIBRARY CHAIN FOR PAYROLL APPLICATIONS
  TYPE: PRODUCTION       FILE: PAYROLL-PROD
                                       MVS
OBJECT LIBRARY ===> payroll-prod-reloc
  LOAD LIBRARY ===> payroll-prod-load
  DBRM LIBRARY ===> DB2 DBRM library
                                       CMS
OBJECT LIBRARY ===>
  LOAD LIBRARY ===>
                                       VSE
OBJECT LIBRARY ===>
  LOAD LIBRARY ===>
```

OBJECT LIBRARY

The library to contain object (or relocatable) modules.

LOAD LIBRARY

The library to contain load modules or phases.

DBRM LIBRARY

For use with DB2 for z/OS and OS/390. The PDS to contain the DB2 for z/OS and OS/390 pre-compile output (DBRMs).

Note: The OBJECT, LOAD, and DBRM LIBRARY fields support selection list processing. Enter ***** or wildcard specification to obtain a selection list of all operating systems libraries that meet the selection criteria. Enter **S** to select a library from the list. Press Enter to return to the Associated Files panel, with the selected library name entered in the appropriate LIBRARY field.

Library Definition Panels LCDF

The Library Definition Panels LCDF defines production, Q/A, test, reject, operating system, and history libraries. For each library type, you must fill out a panel to supply library information common to all operating system environments and also, if applicable, information unique to the VM/ESA and VSE/ESA environments. Each library type panel is shown on the following pages.

Production Library

```
                  PRODUCTION LIBRARY

COMMAND ===>

    LIBRARY NAME:  PAYROLL-PROD

                        COMMON INFORMATION

DATA SET NAME ===> acct.payroll.prod.mast        ACCESS ===> R
 HISTORY FILE ===> payroll-hist
     MCD BASE ===> 0000      TYPE ===> BDAM

                         CMS INFORMATION

OWNER===> mvssa  ADDRESS===> 452  MULTIWRITE PASSWORD===> mmvsa

                         VSE INFORMATION

VOLUME ID ===> vselb1
```

Q/A Library

```
                     Q/A LIBRARY

COMMAND ===>

    LIBRARY NAME:  PAYROLL-STAGING

                        COMMON INFORMATION

DATA SET NAME ===> acct.payroll.staging.mast    ACCESS ===> R
     MCD BASE ===> 0000       TYPE ===> BDAM

                        CMS INFORMATION

OWNER ===>    ADDRESS ===>     MULTIWRITE PASSWORD ===>

                        VSE INFORMATION

VOLUME ID ===>
```

Test/Reject Library

```
  TEST/REJECT LIBRARY

COMMAND ===>

LIBRARY NAME: PAYROLL-REJECT


DATA SET NAME ===> acct.payroll.reject.mast
MCD BASE ===> 1111 TYPE ===> BDAM
ADD LIBRARY MODIFIER ===> EXISTING LIBRARY MODIFIERS: NO

CMS INFORMATION

OWNER===> mvssa ADDRESS===> 452 MULTIWRITE PASSWORD===> mmvsa

VSE INFORMATION

VOLUME ID ===> vselb1
```

Operating System Libraries

```
                OPERATING SYSTEM LIBRARIES
COMMAND ===>

     LIBRARY NAME:   PAYROLL-QA-LOAD

                           COMMON INFORMATION

BACKUP LIBRARY ===>
ADD LIBRARY MODIFIER ===>       EXISTING LIBRARY MODIFIERS: NO

                           MVS INFORMATION

DATA SET NAME ===> acct.payroll.qatest.load

                           CMS INFORMATION
OWNER ===>        MODE ===>        ADDRESS ===>        PASSWORD ===>
         FILE NAME ===>                 FILE TYPE ===>

                           VSE INFORMATION

LIBRARY ===>        SUBLIBRARY ===>
```

History Library

```
                      HISTORY LIBRARY
COMMAND ===>

    LIBRARY NAME:  PAYROLL-HIST

                      COMMON INFORMATION

DATA SET NAME ===> acct.payroll.history.mast
  LOCK STATUS ===> 2
     MCD BASE ===> 3333      TYPE ===> BDAM

                      CMS INFORMATION

OWNER===>mvssa        ADDRESS===>452        MULTIWRITE PASSWORD===>mmvs

                                          VSE INFORMATION
VOLUME ID ===> vselb1
```

The panel for each library type contains a subset of the fields that follow.

For All Operating Systems

Common information applying to all operating systems:

DATA SET NAME

The real data set name for the file.

Note: You can substitute the user ID (or the CCF ID if the user ID was converted in Option 12.8) anywhere in the data set name by coding the @CCFUID@ variable. This is a useful technique for permitting each programmer to own his own test data set. The @CCFUID@ variable is supported for the test and reject master files and their associated object and load libraries, the VSE/ESA sublibraries, and the VM/ESA owner and password fields. The @CCFUID@ variable is not supported for the production and Q/A master files and their associated object and load libraries.

ACCESS

Defines the type of access allowed for production or Q/A libraries from ELIPS when you select the CCFCHK ELIPSGEN parameter, and from CA Roscoe when you select the LIBRCCF initialization option. LIB/CMS does not use this facility. In VM/ESA, you can restrict access as necessary through the use of LINK passwords.

You must restart CA Roscoe to activate changes to the ACCESS field.

A

Any

B

Browse

- For CA Roscoe, ATTACH and PRINT only
- For ELIPS, BROWSE, PRINT, and INFO functions only, without INFO update capabilities

R

(Default) Read, Browse, and Copy

- For CA Roscoe, IMPORT, ATTACH, and PRINT.
- For ELIPS, BROWSE, PRINT, COPY, and INFO, without INFO update capabilities. **N** None.

BACKUP LIBRARY

The file name (real or symbolic) for the backup load library. Does not apply for object libraries.

HISTORY FILE

The file name (real or symbolic) for the history master file associated with this production file.

Note: The BACKUP LIBRARY and HISTORY FILE fields support selection list processing. Enter ***** or wildcard specification to obtain a selection list. For the BACKUP LIBRARY, all operating system files that meet the selection criteria display. For the HISTORY FILE, all history master files that meet the selection criteria display. Enter **S** to select a library from the list. Press Enter to return to the Operating System Files panel, with the selected library name entered in the appropriate field.

MCD BASE

The MCD *base* code for the master file (supply only the base code, *not* the date plus the base code). See *the Security Administration G*uide for more information on the MCD.

TYPE

The default is BDAM. VM/ESA is valid for master files initialized with the LIB/CMS LIBMAST command. Specify BDAM for z/OS and OS/390 BDAM, PDS, and PDS/E master files.

ADD/DELETE LIBRARY MODIFIER

Enter **A** or **Y** to add a modifier. Enter **D** to delete the displayed modifier.

A library modifier provides a way to override information supplied for a test master file, reject master file, or test operating system library. This is especially useful when defining data sets with the @CCFUID@ variable as you might want to define user specific information about the actual data set.

For example, if each programmer has his own test data set, you could define LIBR.@CCFUID@.TEST as the test master file data set name. This way, when programmer JONES performs a logout, LIB/CCF copies the member into LIBR.JONES.TEST. Similarly, if programmer SMITH performs a logout, the member is copied into LIBR.SMITH.TEST. But what if programmer SMITH wants to use an MCD base different from the MCD base specified for LIBR.@CCFUID@.TEST? The solution is to supply a library modifier for SMITH that specifies the correct MCD base for LIBR.SMITH.TEST.

Another example is a VM/ESA site requiring data sets to reside on different volumes. This scenario requires CCF to provide different link information (owner, address, and password) that could be supplied by specifying a library modifier.

EXISTING LIBRARY MODIFIERS

(Display only.) YES indicates that modifiers exist. NO indicates that no modifiers exist. Use the up and down PF keys to scroll forward and backward, respectively.

CCF USER ID

The actual user ID (or the CCF ID if the user ID is converted in Option 12.8) for which the library modifier is supplied.

For VM/ESA

The VM/ESA information (required only for VM/ESA sites):

OWNER

The machine that owns the file. Alternatively, you can specify the @CCFUID@ variable if the machine ID matches the user ID.

ADDRESS

For master files only. The real address of the file.

MULTIWRITE PASSWORD

For master files only. The VM/ESA multiwrite password for the file. Alternatively, you can specify the @CCFUID@ variable if the password matches the user ID.

ADDR

For operating system files only. The real address of the file.

MODE

For operating system files only. A valid VM/ESA link mode.

PASSWORD

For operating system files only. The correct password for the MODE specified.

FILE NAME

(Optional.) For operating system files only. Can specify the file name of an operating system library (for example, OS simulated load libraries or macro libraries).

FILE TYPE

(Optional.) For operating system files only. Can specify the file type.

For VSE/ESA

The VSE/ESA information (required only for VSE/ESA sites):

VOLUME ID

The volume for the file.

LIBRARY

The VSE/ESA library name.

SUBLIBRARY

The VSE/ESA sublibrary name. Alternatively, you can specify the @CCFUID@ variable if the sublibrary name matches the user ID.

When the last panel is filled in, LCDF has all the necessary information to create a library chain, and does so. You can enter a command of **FINISH** at any time to bypass unneeded panels.

Modify a Library Chain

When you select Option 12.11.2 (Modify Library Chain), the Chain Selection Panel appears.

```
                              SELECT CHAIN
COMMAND ===>

   CHAIN NUMBER ===>      (blank or asterisk for selection list)
```

Enter a chain number. Alternatively, you can enter ***** or blank to obtain a selection list of all the chains. Enter **S** to select a chain. The display that follows begins with the same panels as that of the create function, starting with the Chain Description Panel.

All of the information is modifiable.

The LCDF screens are presented in the following order.

1. Chain Selection
2. Chain Description
3. Chain Structure
4. Chain Attributes
5. Associated Files
6. Library Information

The Library Information panel appears only for new files that you added to the chain through the Chain Structure panel.

Note: Do not alter chains until all related outstanding members are processed to completion, or the logout has been deleted. Changing chains when members are outstanding can cause unpredictable results.

Modify the Library Definition

When you select Option 12.11.3 (Modify library definition), the Library Selection Panel appears.

```
                        SELECT LIBRARY
COMMAND ===>

   LIBRARY NAME ===>
                     (blank or asterisk for selection list)
```

Enter the name of the library to modify. Alternatively, wildcard selection is available. Enter *S* to select one of the libraries from the selection list. The appropriate Library Information panel then appears, depending on whether the library you selected is a PROD, Q/A, TEST/REJ, OPERATING SYSTEM, or HISTORY library. Modify the information on the panel as necessary.

Deactivate a Library Chain

When you select Option 12.11.4 (Deactivate library chain), the Chain Selection Panel appears.

```
                          SELECT LIBRARY
COMMAND ===>

   LIBRARY NAME ===>

                        (blank or asterisk for selection list)
```

Enter the chain number of the chain to deactivate. Alternatively, enter an asterisk or blank to obtain a selection list of all chains. Use an S to select a chain. The Deactivate Chain confirm panel displays.

```
                          DEACTIVATE CHAIN
COMMAND===>

            NUMBER: 00001                    GROUP: PAYROLL

DESCRIPTION: LIBRARY CHAIN FOR PAYROLL APPLICATIONS

CONFIRM===>     (enter “Y” to confirm)
```

The deactivate library chain function lets you deactivate a chain. Once a library chain is deactivated, you can no longer use it to logout a member. LIB/CCF continues to process logged out members still outstanding at the time the library chain is deactivated. You can use Option 12.11.5 (Delete Unused Chains/Library Definitions) to physically delete all deactivated chains. A chain is not deleted until all related outstanding members are processed to completion.

Delete Deactivated Chains

When you select Option 12.11.5, all deactivated chains and library definitions are deleted. The Library/Chain Statistics panel appears, displaying the summary information.

```
                        LIBRARY/CHAIN STATUS
COMMAND===>

DELETED:
 CHAIN ENTRIES: 0000001        LIBRARY DEFINITIONS: 0000003
        CHAINS: 00001 LIBRARY DEFINITION MODIFIERS: 0000000

PENDING DELETION:
 CHAIN ENTRIES: 0000000                       CHAINS: 00000

ACTIVE:
CHAIN ENTRIES: 0000002         LIBRARY DEFINITIONS: 0000006
       CHAINS: 00001  LIBRARY DEFINITION MODIFIERS: 0000000
```

The Library/Chain Statistics panel shows the number of deleted, deactivated, and active chains. The deactivated total is the total number of chains that were deactivated but cannot be deleted because they are still in use.

Library Chain Definition Report (Option 8.9)

The Library Chain Definition Report generates several different formats of chain or library definition reports. Only the system administrator can request this report. The following screen shows the panel you use to generate the Library Chain Definition Report.

Enter the report type.

If you are requesting a Library Definition Report, enter a library name to request the report for one library. If you do not, the report is for all library definitions.

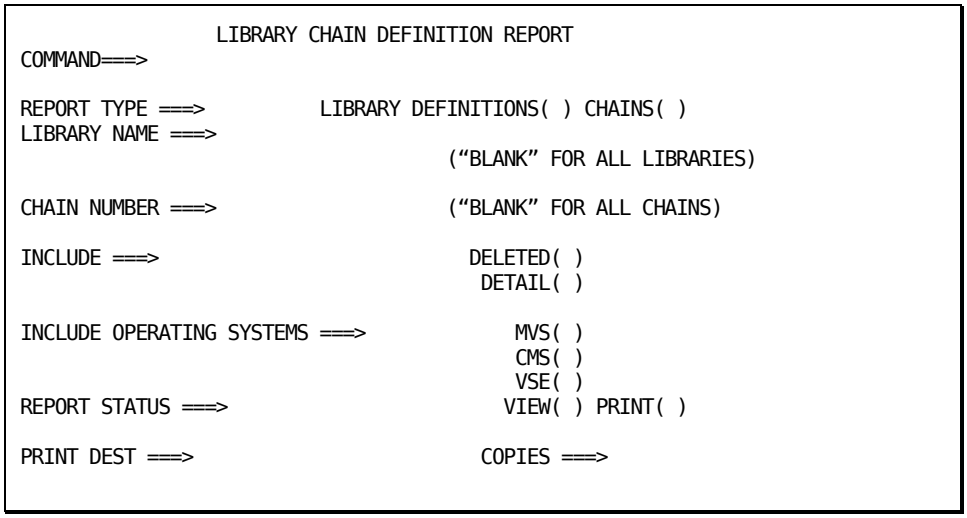
If you are requesting a Chain report, enter a chain number to request the report for one chain. If you do not supply a chain number, the report is for all chains.

You can select the DETAIL option for a chain report and include the chain attribute information for each entry in the chain. When you select the DETAIL option, you can also select options to include deactivated entries (DELETED) and specific operating systems information (INCLUDE OPERATING SYSTEMS).

Check the report status you want. The choices are VIEW and PRINT. If you choose to view the report online, you can also print it from the viewing panel with the PRINT command issued from the command line. If you choose PRINT, fill in the PRINT DEST data field to print the report on a printer other than the default printer. Enter the number of copies in the COPIES data field if you want to print more than one copy.

For LIB/CCF CA Roscoe and ISPF(TSO), the report prints either by the available online print function (RPS for CA Roscoe, for example) or by a batch job, depending on what you specified for the PRINT keyword in Option 12.0 (PROFILE).

For LIB/CCF ISPF(VM/ESA), issue the PRINT command to display the print specification panel.



Group Definition Table (Option 12.12 Except AllFusion CA-Vollie)

The Group Definition Table defines a group of users to LIB/CCF. Once defined in this table, you can use the Group ID in the following LIB/CCF administrator tables:

- User Definition table (Option 12.1) for USER ID and USER Key.
- Manager Definition table (Option 12.2) for MGRS ID.
- Programmer Definition table (Option 12.3) for MGRS ID and PGMR ID.

If you assign a change request to a Group ID, everyone defined in that group can work on that change request.

You can use the Group ID where the CCFID is used in other Option 12 tables. A Group ID cannot be a CCFID, nor can it include a Group ID.

Group Definition Table Panel Fields

The panel fields are:

GROUP ID

Specify the name of the group. You cannot change or delete this field if the Group ID exists in any of the LIB/CCF tables listed above or if the Group ID is used in any outstanding change requests.

MEMBER ID

Specify the members of the group.

Group Definition Table Panel Commands

The commands available from this panel are:

=n (jump)

The value of *n* is an option number. Processes end before the jump (updates the table).

CANCEL

Ends processing and *does not* update the table.

END PF key

Ends processing and updates the table.

FIND string

Positions the display to the first occurrence of the specified string. Do not use delimiters.

NEXT

(CA Roscoe.) Positions the display to the next occurrence of the string that was previously specified by the FIND command.

RFIND

(TSO and VM/ESA.) Positions the display to the next occurrence of the string that was previously specified by the FIND command.

SORT [GROUP|MEMBER]

Sorts the table by either the GROUP ID or MEMBER ID. The table is automatically sorted when saved. Specified with no operand, the table sorts by GROUP ID.

Chapter 3: LIB/CCF Batch Utilities for CA Roscoe and ISPF(TSO)

The LIB/CCF system provides seven batch programs.

- Change request status report ($CCFB101).
- Logout utility ($CCFB102).
- Login utility ($CCFB109).
- Change request initiation and assignment utility ($CCFB103).
- Change request close utility ($CCFB104).
- Source to load synchronization report utility (LIBRSCAN).
- Current archive level copy utility (LIBRCOPY).

This section contains the following topics:

Change Request Status Report ($CCFB101) (see page 79)
Batch Logout Utility ($CCFB102) (see page 81)
Batch Login Utility ($CCFB109) (see page 84)
Batch Change Request Initiation ($CCFB103) (see page 92)
Batch Change Request Close ($CCFB104) (see page 97)
LIBRSCAN (see page 100)
LIBRCOPY (see page 101)

Change Request Status Report ($CCFB101)

You can use the Change Request Status Report ($CCFB101) program to print comprehensive summary reports on all:

- Change requests
- Open change requests
- Unassigned change requests
- Closed change requests.

This report can list the change requests for a single programmer, manager, or application system. It optionally prints the analysis section for closed change requests and the status of logouts, links, moves, and batch jobs associated with each selected change request.

Use the following JCL for the report:

```
//REPORT JOB ,pgmr,CLASS=X
//STEP EXEC PGM=$CCFB101,PARM='selected parms'
```

The default SYSOUT characteristics for the report are:

```
CLASS=A,DEST=LOCAL,COPIES=1
```

You can change the SYSOUT characteristics with a DD statement for CCFPRINT in that JCL and defining the SYSOUT:

```
//REPORT JOB ,pgmr,CLASS=X
//STEP EXEC PGM=$CCFB101,PARM=ALL
//CCFPRINT DD SYSOUT=1,COPIES=3
```

If the program library is not in a LNKLST, you must include a STEPLIB to the LIB/CCF load library containing $CCFB101 and $CCFCOMI.

All necessary files are allocated dynamically. The LIB/CCF system master file is dynamically allocated based on the SYSMAST keyword of the $CCFGEN macro.

Parameters

The parameter options are:

type

Specifies the type of change request to report. Specify one of the following:

OPEN

(Default) All open change requests.

UNASSIGNED

All unassigned change requests.

CLOSED

All closed change requests.

ALL

All open and closed change requests.

ANALYSIS

Prints the analysis section for closed change requests. You can specify this parameter only with ALL or CLOSED. The default is not to print the analysis section.

APPL=[*application name*]

Prints only the change requests for this application system. The default is all applications.

MGR=[*managername*]

Prints only the change requests for this applications manager. The default is all managers.

NOSEP

Suppresses skipping to the top of the page for each change request. The default starts each change request on a new page.

PGMR=[*pgmrname*]

Prints only the change requests assigned to this programmer. The default is all programmers.

STATUS

Prints the status of logouts, links, moves, and batch jobs associated with all selected change requests. The default does not print the status.

Batch Logout Utility ($CCFB102)

A batch logout utility ($CCFB102) is available that allows for logouts of more than one member at the same time. You can use the CA Librarian Group Processing Option (GPO) to generate the SYSIN statements. See the *System Services Guide* for details on GPO. The JCL requires only an EXEC statement and a SYSIN file of control statements as all other files are dynamically allocated.

If the program library is not in a LNKLST, you must include a STEPLIB to the LIB/CCF load library containing $CCFB102 and $CCFCOMI.

All necessary files are allocated dynamically. The LIB/CCF system master file is dynamically allocated based on the SYSMAST keyword of the $CCFGEN macro.

Condition Codes

$CCFB102 issues the following condition codes:

No errors occurred during execution.

(Warning) The LIB/CCF administrator selected the multiple module logout option (ALLOWMULT YES) and at least one module that was logged out during this execution is currently outstanding to another user. No other errors were encountered. See the CCF Batch Logout Activity report for warning messages.

At least one error occurred. If you specified the parm option UNCONDITIONAL and an error occurs, processing continues for all logouts. If you specified the parm option CONDITIONAL, processing terminates with the first error. See the CCF Batch Logout Activity report for error messages.

Parameters

The parameter options are:

[UNCONDITIONAL|CONDITIONAL]

Specifies whether processing terminates if an error occurs.

UNCONDITIONAL

(Default) Logs out all modules that can be logged out. You can abbreviate this parameter to UNCOND.

CONDITIONAL

Terminates processing if an error occurs (for example, if a requested module is already logged out to another programmer). You can abbreviate this parameter to COND.

SYSTEM=[ISPF|ROSCOE|VOLLIE|CMS]

Identifies which LIB/CCF system tables are used for the logout. ISPF is the default.

SYSIN Control Statements

The SYSIN control statements are:

-COPY [modname]

Specifies the module name to log out

-CRID WOnnnnnn

Specifies the Change Request ID for the logout.

-CUID [id]

Specifies the LIB/CCF ID of the user performing the logout (specify -CUID when the change request is assigned to a LIB/CCF group).

-MFID ***nnn***

Specifies the master file ID. The programmer assigned to the change request must obtain this ID from the Option 4 MODULE LOGOUT panel. The ID appears on that panel to the left of the production/test master file pair, under the field Available Files.

-LANG ***xxxxxxxx***

Specifies the valid language code to use for history member creation. Can be one to eight characters. This language determines what action takes place when a module is logged in. For example, specifying a language of COB might compile the module at login.

-DESC[description]

Specifies the module description to use for history member creation. Can be up to 30 characters in length.

-APPL[appl sys]

Specifies the application system to use for history member creation. Can be up to 20 characters in length.

-ABST[text of abstract record]

Specifies the abstract for history member creation. It can be 1 to 16 continuous records. The text supplied for each abstract record can be up to 61 characters in length.

-COM[comment]

Specifies that comments appearing after the -COPY statement print before the -COPY statement on the report.

-EMOD

(Optional) Marks the end on one change request's (-CRID) control SYSIN Control Statement.

-END

(Optional) Marks the end of the entire control stream.

A -COPY statement is required for each module to log out. You must specify a -CRID, -MFID, -LANG, -DESC, -APPL, and at least one -ABST statement as follows:

- You can place these statements before or after the *first* -COPY statement, in any order. Once you specify one of these values, it remains in effect for subsequent -COPYs until another control statement of the same type changes it.
- For any -COPY statement that is not the first -COPY statement, you must place these statements after the -COPY statement. They affect that -COPY and subsequent -COPY statements until another control statement of the same type changes it.
- The -LANG, -DESC, -APPL, and -ABST control statements are required for program execution. However, the information supplied on these control statements is used only for history member *creation***.** If the history member already exists, the information is ignored.

Examples

The following JCL example executes $CCFB102:

```
//LOGOUT JOB,ACCT,PGMR,CLASS=X
//STEP EXEC PGM=$CCFB102,PARM='COND,SYSTEM=ROSCOE'
-MFID 1
-CRID WO000005
-LANG ASM

-DESC VENDOR PAYROLL SYSTEM
-APPL PAYROLL
-ABST VENDOR PAYROLL SYSTEM MEMBER
-COPY PAY01
-COM PROGRAM MEMBERS:

 -COPY PAY02
-COPY PAY03
-ABST VENDOR PAYROLL SYSTEM JCL MEMBER
-COPY PAYJCL
-COM  JCL MEMBERS:

 -COPY PAYJCL2
```

The SYSOUT characteristics for the report are:

```
DEST=LOCAL,COPIES=1,CLASS=A
```

(or as specified on the SYSOUT parameter in Option 12.0).

You can change the SYSOUT characteristics by including a DD statement for CCFPRINT in the JCL and defining the SYSOUT.

```
//LOGOUT JOB,ACCT,PGMR,CLASS=X
//STEP EXEC PGM=$CCFB102,PARM=COND
//CCFPRINT DD SYSOUT=W,COPIES=3
  -COPY modname
     .    .
     .    .
```

Batch Login Utility ($CCFB109)

When you use the LIB/CCF Option 5 and Option 9 Group Processing Option SUBMIT function, LIB/CCF submits a batch job that executes a batch ISPF login utility program. A batch job can also execute this program ($CCFB109) outside of LIB/CCF.

$CCFB109 lets you login more than one member at the same time.

$CCFB109 is an ISPF batch program. As it executes, it can submit other batch jobs, using ISPF JCL skeletons, to perform the requested functions (for example, a compile or a move). The ISPF JCL skeletons normally obtain job card, job scheduling, and DB2 for z/OS and OS/390 subsystem information from an online LIB/CCF system. However, when the skeletons are submitted by $CCFB109, $CCFB109 SYSIN control statements provide this information.

Condition Codes

$CCFB109 issues the following condition codes:

No errors occurred during execution.

See the Group Processing Activity report for error messages ($CCFB109 automatically generates this report).

20

The program executed in a non-ISPF batch environment.

Group Processing Activity Report

The following report shows a sample of the Group Processing Activity report.

```
*--------------------------- LIB/CCF GROUP PROCESSING ACTIVITY (BATCH EXECUTION) ----------------------------------*
*** SYSTEM=LIB/CCF-ROSCOE ***
*** LOGIN (OPTION 5) MODE ***
*** SUBMIT CROSS-REFERENCE FOR VERSION FAILURE ***
*** EXITS ARE DISABLED ***

JC //CCFJOB   JOB (ACCOUNTING),PGMR02.BU03,CLASS=0,                    0000004
JC //             MSGCLASS=T                                           0000004
CCFID FRAJO03                                                         0000004
HISTORY LOGIN FOR PAYROLL APPLICATION                                 0000004
S 5                                                                   0000005
   Movement Complete         CC008CI:     The movement of PKRMETH was completed successfully
      1 module(s) processed  CCF008FI:    1 module(s) were either PROCESSED, DELETED, or REJECTED
```

Note: You can find messages displayed in the report in the *Messages Guide.*

Parameters

All parameters are optional. The parameters are:

SYSTEM=[ISPF|ROSCOE]

Identifies which LIB/CCF system tables are used for the login. ISPF is the default.

PGMR

Enables the same processing as LIB/CCF Option 5 (LOGIN), whereby the programmer performs the login. If omitted, processing is the same as LIB/CCF Option 9 (MOVE), whereby the control group processes the programmer's login request.

OVR

Overrides all version failures. Replaces members in the target library.

OVR0F

Overrides only the first version failure.

SKP

Executes a Cross-Level Comparison job when a version failure occurs. Processing continues.

SKP0F

Executes a Cross-Level Comparison job only when the first version failure occurs. Processing stops if another version failure occurs.

NXT

Bypasses any LIB/CCF exit processing. This is useful to bypass a login exit that performs online activity, such as displaying a panel.

SYSIN Control Statements

The SYSIN control statements are:

CCFID ***userid***

Specifies the LIB/CCF ID of the user executing the job. Required only if no access control facility (such as CA Top Secret, CA ACF2, or RACF) is active.

JC job statement

Specifies the 1- to 4-line job card statement to use in jobs that $CCFB109 submits. Each line can be up to 61 characters.

DSN ***dsnid***

(Optional) Specifies the 1- to 4-character DB2 for z/OS and OS/390 subsystem name (specified in LIB/CCF Option 0.2).

HISTORY update data

(Optional) Specifies 1 to 19 lines of history update information. Each line can be up to 61 characters long. If omitted, $CCFB109 generates a generalized history card indicating the date and time of the login, and that the login was performed in batch. The history update information applies to all subsequent functions until respecified.

SCHEDULE ***[mmddyyhhmmss|****] ***[predname]***

(Optional) Specifies job scheduling information for jobs that $CCFB109 submits. The scheduler information applies to all subsequent functions until respecified. See the "Scheduling z/OS and OS/390 Batch Jobs in LIB/CCF" chapter for details on scheduling jobs through LIB/CCF.

mmddyyhhmmss

Specifies the date and time the job should execute. Specify an asterisk (*) for the current date and time. If you specify SCHEDULE without the date and time parameter, job scheduling for subsequent jobs terminates.

predname

pecifies the name of the job that must execute before this job (that is, the predecessor job).

SELECTx ***WOnnnnnn***

Specifies the type of selection to perform, indicated by the value of *x* (an optional letter). If you specify PGMR, you can specify F. If you omit PGMR (and control group processing is in effect), you can specify letters F, C, L, or U. See the *LIB/CCF User Guide* for a complete description of the SELECTx command. The value of *nnnnnn* is the change request number.

DESELECT ***[library[***(***member***)]]

(Optional) Specifies the library and member to remove generated SELECTx processing from the selection list.

function ***[library[***(***member***)]]

Specifies the function to perform during the login. If you specify PGMR, valid functions are PROCESS, DELETE, TEST, and SUBMIT. If you omit PGMR (and control group processing is in effect), valid functions are COMPILE, PROCESS, MOVE, REJECT, SUBMIT, and UNLOCK. See the *LIB/CCF User Guide* for a description of these functions.

REFRESH

Rebuilds the selection list.

ISPF Execution

Use the following sample JCL to execute $CCFB109.

```
//CCFJOB   JOB (ACCOUNTING),PGMR02.BU03,CLASS=0,
 //             MSGCLASS=T
//CREPROF  EXEC PGM=IEFBR14
//O        DD  DSN=&&TEMPPROF,DISP=(,PASS),UNIT=VIO,
 //             SPACE=(3120,(30,,2)),
 //             DCB=(DSORG=PO,RECFM=FB,LRECL=80,BLKSIZE=3120)
 //         EXEC PGM=IKJEFT01,DYNAMNBR=25,REGION=2000K,PARM=
//ISPLLIB  DD  DISP=SHR,DSN=CAI.CALJLINK           A       LIBR LOAD LIB
//ISPPLIB  DD  DISP=SHR,DSN=CAI.CALJPENU         B       LIBR PANEL LIB
//         DD  DISP=SHR,DSN=ISR.V3R5M0.ISRPENU           ISR PANEL LIB
//         DD  DISP=SHR,DSN=ISP.V3R5M0.ISPPENU           ISP PANEL LIB
//ISPMLIB  DD  DISP=SHR,DSN=CAI.CALJMENU         C       LIBR MESSG LIB
//         DD  DISP=SHR,DSN=ISR.V3R5M0.ISRMENU           ISR MESSG LIB
//         DD  DISP=SHR,DSN=ISP.V3R5M0.ISPMENU           ISP MESSG LIB
//ISPSLIB  DD  DISP=SHR,DSN=CAI.CALJSENU         D       LIBR SKEL LIB
//         DD  DISP=SHR,DSN=ISR.V3R5M0.ISRSENU           ISR SKEL LIB
//         DD  DISP=SHR,DSN=ISP.V3R5M0.ISPSLIB           ISP SKEL LIB
//ISPTLIB  DD  DISP=SHR,DSN=CAI.CALJTENU         E       LIBR TABL LIB
//         DD  DISP=SHR,DSN=ISR.V3R5M0.ISRTLIB           ISR TABL LIB
//         DD  DISP=SHR,DSN=ISP.V3R5M0.ISPTENU           ISP TABL LIB
//SYSPROC  DD  DISP=SHR,DSN=ISR.V3R5M0.ISRCLIB   F       ISR CLIST LIB
//ISPPROF  DD  DSN=&&TEMPPROF,DISP=(SHR,PASS)
 //ISPLOG   DD  SYSOUT=K,DCB=(DSORG=PS,RECFM=VA,LRECL=125,BLKSIZE=129)
 //ISPCTL1  DD  DISP=(,DELETE),UNIT=VIO,
 //             SPACE=(3120,(30,5)),
 //             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120)
 //SYSTERM  DD  SYSOUT=*
//SYSTSPRT DD  SYSOUT=*
//SYSPRINT DD  SYSOUT=*
//CCFPRINT DD  SYSOUT=*
//SYSTSIN  DD  *
ISPSTART PGM($CCFB109) NEWAPPL(CCF@) BDISPMAX(99999) +
    PARM(+               }
         SYSTEM=ROSCOE,+ }
         PGMR,+          } G
         NXT+            }
          )              }
//SYSIN    DD  *
JC //CCFJOB   JOB (ACCOUNTING),PGMR02.BU03,CLASS=0, }
JC //             MSGCLASS=T                        }
CCFID PGMR02                                       }
HISTORY LOGIN FOR PAYROLL APPLICATION              }
S 5                                                }
PROCESS                                            }
S 6                                              H }
TEST                                               }
HISTORY LOGIN FOR ACCOUNTS REC APPLICATION         }
S 7                                                }
PROCESS                                            }
```

- A—Specify the CA Librarian load library containing the LIB/CCF load modules, including $CCFCOMI.
- B—Specify the ISPF panel library.
- C—Specify the ISPF message library.
- D—Specify the ISPF skeleton library.
- E—Specify the ISPF table library.
- F—Specify the ISPF CLIST library.
- G—Specify the $CCFB109 execution parameters described earlier in this section.
- H—Specify the $CCFB109 SYSIN control cards described earlier in this section.

All necessary LIB/CCF files are allocated dynamically. The LIB/CCF system master file is dynamically allocated based on the SYSMAST keyword of the $CCFGEN macro.

CA Roscoe Execution

Because $CCFB109 is an ISPF program that tailors ISPF JCL skeletons, CA Roscoe sites must use RPF CCFCBHSK to create ISPF JCL skeletons from the JCL skeletons stored on the LIB/CCF CA Roscoe key. You should also execute RPF CCFCBHSK whenever you update an CA Roscoe skeleton.

Perform the following steps to execute the RPF.

1. Customize the LIB/CCF Option 12 administrator tables if they are not already customized.
2. Customize JCL skeleton $CHGJ500 on the LIB/CCF CA Roscoe key. Specifically, modify the Computer Associates libraries and the ISPF libraries to correctly reflect the libraries used at your site.

3. Execute CCFCBHSK using the following parameters:

   DS(*target.ispf.skeleton.library*)

   (Optional) Specifies the data set name of the ISPF skeleton library to contain the skeletons that RPF generates. If omitted, CAI.CLJ41CSL is used as the target ISPF skeleton library.

   PFX

   (Optional) Specifies the prefix of the CA Roscoe key where LIB/CCF is installed. PFX is needed only if the RPF is executed from a key other than where LIB/CCF is installed. LANG*(language*)—(Optional) Specifies the language the RPF is to process (that is, ISPF skeletons are generated for skeletons with this language). The language must be defined in the Option 12.5 (LANG) table.

   If omitted, all skeletons defined in the Option 12.5 (LANG) table are processed to create corresponding ISPF skeletons.

   SKEL(*skeleton*)

   (Optional) Specifies a single skeleton to convert. This parameter is useful when you modified a single CA Roscoe skeleton and want to update only that skeleton in the ISPF skeleton library.

   NOREP

   (Optional) Indicates that existing skeletons are not replaced in the ISPF skeleton library.

Example: Add Skeletons

This example adds all skeletons with a language of DAT and that are on the CA Roscoe key with a prefix of CCF, to the ISPF skeleton library with a data set name of CAI.LIBR.CCF.ISPSLIB unless they already exist in the library.

```
EXEC CCFCBHSK DS(CAI.LIBR.CCF.ISPSLIB) PFX(CCF) LANG(DAT) NOREP
```

Once the skeletons exist in the ISPF skeleton library, $CCFB109 can execute using the JCL listed under ISPF Execution earlier in this chapter.

Batch Change Request Initiation ($CCFB103)

The Batch Change Request Initiation program ($CCFB103) opens and assigns, or just assigns, change requests. It is available for batch execution, or another program can call it as an interface vehicle. The JCL requires only a program execute statement and SYSIN statements as all needed files are dynamically allocated. The LIB/CCF system master file is dynamically allocated based on the SYSMAST keyword of the $CCFGEN macro.

```
//OPEN JOB ,pgmr,CLASS=X
//STEP EXEC PGM=$CCFB103
control statements
      .
      .
      .
```

Parameters

The parameter options for Batch Change Request Initiation are:

[UNCONDITIONAL|CONDITIONAL]

Specifies whether to continue processing in case of error.

UNCONDITIONAL

(Default) Processes all verbs that can be processed. You can abbreviate this parameter to UNCOND.

CONDITIONAL

Terminates processing if an error occurs. You can abbreviate this parameter to COND.

SYSTEM=[ISPF|ROSCOE|VOLLIE|CMS]

Identifies which LIB/CCF system tables are used for the logout. ISPF is the default.

SYSIN Control Statements

The SYSIN control statements are as follows:

Required for OPEN

-USER *ccfid*

Specifies the CCFID authorized for the OPEN, one to eight characters.

-TYPE[PROBLEM|MODIFICATION]

Specifies the change request type.

-RNME *name*

Specifies the requestor's name, up to 20 characters in length.

-RPH# *number*

Specifies the requestor's phone number, up to 20 characters in length

-RMGR *name*

Specifies the requestor's manager's name, up to 20 characters in length.

-RDPT *name*

Specifies the requestor's department name, up to 20 characters in length.

-APPL *appl sys*

Specifies the application system the user is authorized to open a change request for, up to 20 characters in length.

-CORD *name*

Specifies the coordinator's name, up to 20 characters in length.

-CPH# *number*

Specifies the coordinator's phone number, up to 20 characters in length.

-DESC *desc*

Describes the change request. The description can be 1 to 10 continuous records. Each description record can be up to 61 characters in length.

-OATH *code*

Specifies the opener's authorization code (if required), one to eight characters.

Required for ASSIGNMENT

-COMM *comment*

Provides the manager's comment, 1 to 3 continuous records. Each comment record can be up to 61 characters in length.

-ANME *ccfid*

Specifies the CCF ID of the programmer to whom the change request is assigned, one to eight characters.

-AATH *code*

Specifies the manager's authorization code (if required), one to eight characters.

CA Netman

-NETM ***id***

Specifies the 16 character CA Netman change ID and task ID. Reserved for CA internal use. See the “LIB/CCF-CA Netman Interface” chapter for details on the LIB/CCF-CA Netman interface.

Processing Verbs

-OPEN *[WOnnnnnn]*

Opens a change request. You can optionally specify the change request number by specifying WO*nnnnnn*, where *nnnnnn* is the number of the change request.

-ASGN *[WOnnnnnn]*

Assigns the change request (if it exists) or opens and assigns the change request if it does not exist when you specify a change request number. Opens and assigns the next change request if you do not specify a change request number.

-COM *comment*

(Optional) Provides documentation.

-EMOD

(Optional) Marks the end of one change request’s control statements.

-END

(Optional) Marks the end of the entire control stream.

You must specify the required control statements as follows:

- You must specify the required statements before the first verb statement, in any order.
- Once you specify one of these values, it remains in effect for subsequent verbs until another control statement of the same type changes it.
- The control statement must precede the verb statement to change the value for that verb statement.

Example

The following example shows a $CCFB103 execution.

```
//OPEN JOB,ACCT, PGMR,CLASS=X
//STEP EXEC PGM=$CCFB103,PARM=COND
-USER PAPERSYS
-RNME H. Schmidt
-RPH# 9100
-RMGR R. Deal

-APPL PAYROLL
-RDPT PAYROLL
-CORD H. Schmidt
-CPH# 9100
-DESC Modify report 10 to include subtotals for each cost
-DESC center.

 -TYPE MODIFICATION
-COM ** REPORT 10 MODIFICATION REQUEST **
-OPEN
-DESC Modify Report 14 to show the current employee address.
 -COM ** REPORT 14 MODIFICATION REQUEST **
-OPEN

-DESC correct report 23 to hold 7 digits in the total fields
-DESC instead of truncating the number.
 -TYPE PROBLEM
-COM ** REPORT 23 TRUNCATION ERROR **
-OPEN
```

The SYSOUT default for the report is:

```
DEST=LOCAL,COPIES=1,CLASS=A
```

(or as specified on the SYSOUT parameter in Option 12.0).

You can change the SYSOUT by including the ddname CCFPRINT in the JCL and defining the SYSOUT. For example:

```
//LOGOUT JOB,ACCT,PGMR,CLASS=X
//STEP EXEC PGM=$CCFB103,PARM=COND
//CCFPRINT DD SYSOUT=A,COPIES=3
      .
 control statements
      .
      .
```

Executing $CCFB103 from Another Program

You can execute the utility from another program through the LINK or ATTACH macro, as follows:

```
{LINK  } EP=$CCFB103,PARAM=(optaddr,cridaddr,altddname),VL=1
{ATTACH}
```

All parameters are optional. You can omit them from the PARAM list from right to left or by specifying 0. For example:

```
LINK EP=$CCFB103,PARAM=(0,CRTABL),VL=1
```

Parameters

The parameters options are:

optaddr

Specifies the address of the options list. The list contains those execution parameters usually specified in the PARM parameter of the EXEC JCL statement.

The option list must begin on a halfword boundary. The first two bytes in the list must contain a binary count of the number of bytes in the remainder of the list. For a null list, the count must be zero.

cridaddr

Specifies the address of an area to hold the eight-byte change request numbers that the execution creates. If one change request is processed, the area can be the minimum eight bytes. If more than one is processed, then each change request ID is placed in the table in the order processed.

altddname

Specifies the address of the alternate ddname for SYSIN. This user-specified ddname is used instead of SYSIN for reading the SYSIN control statements.

Batch Change Request Close ($CCFB104)

The batch change request close program ($CCFB104) closes change requests. It is available for batch execution or another program can call it as an interface vehicle. The JCL requires only a program execute statement and SYSIN statements, as all needed files are dynamically allocated. The LIB/CCF system master file is dynamically allocated based on the SYSMAST keyword of the $CCFGEN macro.

```
//CLOSE JOB,ACCT,PGMR,CLASS=X
//STEP EXEC PGM=$CCFB104
control statements
      .  .  .
      .  .  .
```

The SYSOUT default for the report is:

```
DEST=LOCAL,COPIES=1,CLASS=A
```

(or the LIB/CCF Option 12.0 SYSOUT specification).

You can change the SYSOUT by including the ddname CCFPRINT in the JCL and defining the SYSOUT. For example:

```
Changing the SYSOUT Characteristics
//LOGOUT JOB ,pgmr,CLASS=X
//STEP EXEC PGM=$CCFB104,PARM=COND
//CCFPRINT DD SYSOUT=A,COPIES=3
. . .
 control statements
      .  .  .
      .  .  .
```

Parameters

The parameter options for Batch Change Request Close are:

[UNCONDITIONAL|CONDITIONAL]

Specifies whether to continue processing in case of error.

UNCONDITIONAL

(Default) Processes all verbs that can be processed. You can abbreviate this parameter to UNCOND.

CONDITIONAL

Terminates processing if an error occurs. You can abbreviate this parameter to COND.

SYSIN Control Statements

The SYSIN control statements are as follows:

Note: These control statements are required.

-USER *ccfid*

Specifies the CCF ID of the programmer owning the change request to close. The CCF ID can be 1 to 8 characters in length.

-ANAL *analysis*

Specifies the analysis of the problem and solution, 1 to 12 continuous cards. Each analysis record can be up to 64 characters in length.

-AFFT *list*

Specifies the routines affected, 1 to 15 continuous cards. Each AFFT record can be up to 62 characters in length.

-SYSM *date*

System tests modified date.

-SYSV *date*

System tests verified date.

-SYST *list*

System tests added/modified, 1 to 3 continuous cards. Each SYST record can be up to 46 characters in length.

-INPT *list*

Specifies new user input, 1 to 3 continuous cards. Each INPT record can be up to 58 characters in length.

-OUPT *list*

Specifies new user output, 1 to 3 continuous cards. Each OUPT record can be up to 57 characters in length.

-DOCC *list*

Specifies documentation changes, 1 to 3 continuous cards. Each DOCC record can be up to 51 characters in length.

Processing Verb

-CLOSE *WOnnnnnn*

(Required) Specifies the Change Request number to close.

-COM *comment*

(Optional) Comments.

-EMOD

(Optional) Marks the end of one change request's control statements.

-END

(Optional) Marks the end of the entire control stream.

You must specify the required control statements as follows:

- You must specify these statements before the first verb statement, in any order.
- Once you specify one of these values, it remains in effect for subsequent verbs until another control statement of the same type changes it.

Example

The following example shows a $CCFB104 execution.

```
//CLOSE JOB ,pgmr,CLASS=X
//STEP EXEC PGM=$CCFB104,PARM=COND
-USER PAPERSYS
-ANAL Add section REP100 to report program PAYR023 to
-ANAL generate sub totals for each cost center.

 -AFFT PAYRG03
-AFFT PAYRG05
-SYSM 03/23/95
-SYSV 03/25/95
-INPT None.

 -OUPT Reports 10 and 12 will no show a cost center sub
-OUPT total.
 -DOCC Documentation updated to show new report 10 and
-DOCC 12 format.
 -CLOSE WO001287

-ANAL Change section REP048 in report program PAYR039 to
-ANAL expand the total fields to 7 digits.
 -AFFT PAYRG9
-OUPT Report 23 will now show 7 digit totals.
 -DOCC Documentation updated to show new report 23 format.
 -CLOSE WO001293
```

Executing $CCFB104 from Another Program

You can execute the utility from another program through the LINK or ATTACH macro, as follows:

```
{LINK } EP=$CCFB104,PARAM=(optaddr,cridaddr,altddname),VL=1
{ATTACH}
```

All parameters are optional. You can omit them from the PARAM list from right to left or by specifying 0. For example:

```
LINK EP=$CCFB104,PARAM=(0,CRTABL),VL=1
```

Parameters

The parameters are defined as follows:

optaddr

Specifies the address of the options list. The list contains those execution parameters usually specified in the PARM parameter of the EXEC JCL statement. The option list must begin on a halfword boundary. The first two bytes in the list must contain a binary count of the number of bytes in the remainder of the list. For a null list, the count must be zero.

cridaddr

Specifies the address of an area to hold the eight-byte change request numbers that the execution closed. If one change request is processed, the area can be the minimum eight bytes. If more than one is processed, then each change request ID is placed in the table in the order processed.

altddname

Specifies the address of the alternate ddname for SYSIN. Uses this user-specified ddname instead of SYSIN for reading the SYSIN control statements.

LIBRSCAN

You can use the LIBRSCAN program to compare a production load library to a production source library to ensure source-to-load correspondence. LIBRSCAN relies on Source-Load Audit Trail (SLAT) variables to perform the comparison. The Systems *Services Guide* describes the installation and use of this program (which you can also use independently of LIB/CCF).

LIBRSCAN is often used with the output exit LIBREXIT or the input exit IEXSLAT, which are both described in the *System Services Guide*, to ensure a standard format for the SLAT variables in CA Librarian members.

For more information on the Source-Load Audit Trail Variables, see the entry on the VAR option in the *Batch Command Reference Guide.* See the Source to Load Member Verification section in the "LIB/CCF Model System" chapter for more information.

LIBRCOPY

LIBRCOPY is a special utility program that you can use to copy the current level of an archived member from one master file to another. It determines whether the member exists on the receiving master file and adds or updates the member accordingly. The *Systems Services Guide* contains information on LIBRCOPY.

Chapter 4: LIB/CCF Batch Utilities for CA Vollie

The LIB/CCF system provides six batch programs:

- Change request status report (CCFB101)
- Logout utility (CCFB102)
- Change request initiation and assignment utility (CCFB103)
- Change request close utility (CCFB104)
- Source to load synchronization report utility (LIBDSCAN)
- Current archive level copy utility (LIBDCOPY)

This section contains the following topics:

Change Request Status Report (see page 103)
Batch Logout Utility (see page 105)
Batch Change Request Initiation (see page 108)
Batch Change Request Close (see page 113)
LIBDSCAN (see page 116)
LIBDCOPY (see page 116)

Change Request Status Report

(Requires VSE/ESA release 1.3 or later.) You can use the Change Request Status Report program to print comprehensive summary reports on all:

- Change requests
- Open change requests
- Unassigned change requests
- Closed change requests.

The report can list the change requests for a single programmer, manager, or application system. It optionally prints the analysis section for closed change requests and the status of logouts, links, moves, and batch jobs associated with each selected change request.

Use the following JCL for the report:

```
// JOB REPORT
// EXEC CCFB101,SIZE=AUTO,PARM='selected parms'
```

All necessary files are allocated dynamically. The LIB/CCF system master file is dynamically allocated based on the SYSMAST keyword of the $CCFGEN macro.

If the program library is not in the permanent LIBDEF chain, you must include a LIBDEF for the LIB/CCF core image library (containing CCFB101 and CCFCOMI).

The report prints in upper or mixed case, depending on the UPPERCASE keyword of Option 12.0.

Parameters

The parameter options are:

type

Specifies whether all requests or only the open, unassigned, or closed requests are reported on.

OPEN

(Default) Selects all open change requests.

UNASSIGNED

Selects all unassigned change requests.

CLOSED

Selects all closed change requests.

ALL

Selects all open and closed change requests.

PGMR=**[*pgmrname*]**

Prints only the change requests assigned to this programmer. The default is all programmers.

MGR=[***managername***]

Prints only the change requests for this manager. The default is all managers.

APPL=[***application name***]

Prints only the change requests for this application system. The default is all applications.

ANALYSIS

Prints the analysis section for closed change requests. You can only specify this option with ALL or CLOSED. The default does not print the analysis.

STATUS

Prints the status of logouts, links, moves, and batch jobs associated with all selected change requests. The default does not print the status.

NOSEP

Suppresses skipping to a new page for each change request. The default starts each change request on a new page.

LINES=***nn***

Specifies the number of lines per page for the printed report. Specify a number between 40 and 99 inclusive. The default is 55.

Batch Logout Utility

A batch logout program (CCFB102) is available that allows for logouts of more than one member at the same time. You can use the CA Librarian Group Processing Option (GPO) to generate the SYSIPT control statements. See the *System Services Guide (VSE/ESA)* for details on GPO.

All necessary files (except for the CA Librarian job stream file) are dynamically allocated. The LIB/CCF system master file is dynamically allocated based on the SYSMAST keyword of the $CCFGEN macro.

CCFB102 requires VSE/ESA SP 2 or above.

CCFB102 requires program QLAMADD to update the LIB/CCF system and history master files. Verify that QLAMADD is installed. See the *Getting Started (VSE/ESA)* for details.

CCFB102 creates a Job Submission Record (JSR) in Option 11 for each logout request for an existing module. A JSR is not created for new modules. The JSR is automatically removed from Option 11 if all the logout step condition codes are zero. If any step ends with a non-zero condition code, the JSR remains in Option 11 so that you can resubmit the job.

Parameters

The parm options are:

[UNCONDITIONAL|CONDITIONAL]

Specifies whether processing terminates if an error occurs.

UNCONDITIONAL

(Default) Logs out all members that can be logged out.

CONDITIONAL

Terminates processing if an error occurs (for example, a requested member is already logged out to another user).

SYSTEM=[VOLLIE|ISPF|ROSCOE|CMS]

Identifies which LIB/CCF system tables are used for the logout.

SYSIPT Control Statements

The SYSIPT control statements are:

-COPY **[*modname*]**

Specifies the module name to log out.

-CRID ***WOnnnnnn***

Specifies the Change Request ID for the logout.

-MFID ***nnn***

Specifies the master file ID. The programmer assigned to the change request must obtain this ID from the Option 4 MODULE LOGOUT panel. The ID appears on that panel to the left of the production/test master file pair, under the field Available Files.

-LANG ***xxxxxxxx***

Specifies the valid language code to use for history member creation. Can be one to eight characters. This language determines what action takes place when a module is logged in. For example, specifying a language of COB might compile the module at login.

-DESC [***desc***]

Specifies the module description to use for history member creation. Can be up to 30 characters in length.

-APPL [***appl sys***]

Specifies the application system to use for history member creation. Can be up to 20 characters in length.

-ABST [***text of abstract record]***

Specifies the abstract for history member creation. Can be one to sixteen continuous records. The text supplied for each abstract record can be up to 61 characters in length.

// ID

Passes the ID statement without modification to the Batch Logout job stream.

-LST

Generates a * $$ LST POWER JECL statement for the Batch Logout job stream. Any data following the -LST is used as the operand portion of the * $$ LST statement.

-COM [***comment***]

Specifies that comments appearing after the -COPY statement print before the -COPY statement on the report.

-EMOD

Optional. Marks the end on one change request's (-CRID) control statements.

-END

Optional. Marks the end of the entire control stream.

Specifications

A -COPY statement is required for each module to log out. You must also specify a -CRID, -MFID, -LANG, -DESC, -APPL and at least one -ABST statement as follows:

- You can place these statements before or after the first -COPY statement, in any order. Once you specify one of these values, it remains in effect for subsequent -COPYs until another control statement of the same type changes it.
- For any -COPY statement that is not the first -COPY statement, you must place these statements after the -COPY statement. They affect that -COPY and subsequent -COPY statements until another control statement of the same type changes it.
- The -LANG, -DESC, -APPL, and -ABST control statements are required for program execution. However, the information supplied on these control statements is used only for history member creation. If the history member already exists, the information is ignored.

Example

```
* $$ JOB JNM=xxxxxxxx,CLASS=A,USER=xxxxxx
* $$ LST CLASS=L
* $$ PUN CLASS=A,DISP=I                              A
// JOB xxxxxxxx
// DLBL JOBSTR,'libr.job.stream',0,SD
// EXTENT SYS00n,xxxxxx,1,0,800,30                   B
// ASSGN SYS008,DISK,VOL=xxxxx,SHR
*
// LIBDEF *,SEARCH=(libccf.lib,librarian.lib)        C
// EXEC CCFB102,SIZE=AUTO
-MFID 1
-CRID WO000001
-LANG ASM
-DESC VENDOR PAYROLL SYSTEM
-APPL PAYROLL
-ABST VENDOR PAYROLL SYSTEM MODULE
-COPY PAY01
-EMOD
-END
/*
/&
* $$ EOJ
```

- A—The POWER * $$ PUN statement must specify DISP=I to return the CCFB102 output (the batch logout job stream) to the input queue for subsequent execution.
- B—The DLBL, ASSGN, and EXTENT statements are required for the batch CA Librarian job stream file. The default file ID and unit number are JOBSTR and SYS008, respectively. However, your site might have changed them.
- C—If a system LIBDEF did not define the LIB/CCF and the batch CA Librarian executable libraries, provide a LIBDEF statement in the JCL.

Batch Change Request Initiation

The Batch Change Request Initiation program (CCFB103) opens and assigns, or just assigns, change requests. It is available for batch execution, or another program can call it as an interface vehicle. The JCL requires only a program execute statement and SYSIPT statements as all needed files are dynamically allocated. The LIB/CCF system master file is dynamically allocated based on the SYSMAST keyword of the $CCFGEN macro.

```
// JOB xxxxxxx
// EXEC CCFB103
   control statements
         .
         .
         .
```

Parameters

The parameter options are:

[UNCONDITIONAL|CONDITIONAL]

Specifies whether to continue processing in case of error.

UNCONDITIONA

(Default) Processes all verbs that can be processed. You can abbreviate this parameter to UNCOND.

CONDITIONAL

Terminates processing if an error occurs. You can abbreviate this parameter to COND.

SYSTEM=[ISPF|ROSCOE|VOLLIE|CMS]

Identifies which LIB/CCF system tables are used for the logout. VOLLIE is the default.

SYSIPT Control Statements

The SYSIPT control statements are as follows:

Required for OPEN:

-USER ***ccfid***

Specifies the CCFID authorized for the OPEN, one to eight characters.

-TYPE[PROBLEM|MODIFICATION]

Specifies the change request type.

-RNME ***name***

Specifies the requestor's name, up to 20 characters in length

-RPH# ***number***

Specifies the requestor's phone number, up to 20 characters in length.

-RMGR ***name***

Specifies the requestor's manager's name, up to 20 characters in length.

-RDPT ***name***

Specifies the requestor's department name, up to 20 characters in length.

-APPL ***appl sys***

Specifies the application system for which the user is authorized to open a change request, up to 20 characters in length.

-CORD ***name***

Specifies the coordinator's name, up to 20 characters in length.

-CPH# ***number***

Specifies the coordinator's phone number, up to 20 characters in length.

-DESC ***desc***

Describes the change request. The description can be 1 to 10 continuous records. Each description record can be up to 61 characters in length.

-OATH ***code***

Specifies the opener's authorization code (if required), one to eight characters.

Required for ASSIGNMENT:

-COMM ***comment***

Specifies the manager's comment, one to three continuous records. Each comment record can be up to 61 characters in length.

-ANME ***ccfid***

Specifies the CCF ID of the programmer to whom the change request is assigned, one to eight characters.

-AATH ***code***

Specifies the manager's authorization code (if required), one to eight characters.

Required for CA Netman:

-NETM ***id***

Specifies the 16 character CA Netman change ID and task ID. Reserved for CA internal use. See the "LIB/CCF-CA Netman Interface" chapter in this guide for more information.

Processing Verbs

The processing verbs are as follows:

-OPEN [***WOnnnnnn***]

(Required) Opens a change request. You can optionally specify the change request number by specifying WO*nnnnnn*, where *nnnnnn* is the number of the change request.

-ASGN [***WOnnnnnn***]

(Required) Assigns the change request number (if it exists) or opens and assigns the change request if it does not exist when you specify a change request number. Opens and assigns the next change request if you do not specify a change request number.

-COM *comment*

(Optional) Provides documentation.

-EMOD

(Optional) Marks the end of one change request's control statements.

-END

(Optional) Marks the end of the entire control stream.

Specifications

You must specify the required control statements as follows:

- You must specify the required statements before the first verb statement, in any order.
- Once you specify one of these values, it remains in effect for subsequent verbs until another control statement of the same type changes it.
- The control statement must precede the verb statement to change the value for that verb statement.

Example

The following example is a CCFB103 execution.

```
// JOB xxxxxxx
// EXEC CCFB103,PARM=COND
-USER PAPERSYS
-RNME H. Schmidt
-RPH# 9100

-RMGR R. Deal
-APPL PAYROLL
-RDPT PAYROLL
-CORD H. Schmidt
-CPH# 9100

-DESC Modify report 10 to include subtotals for each cost
-DESC center.
 -TYPE MODIFICATION
-COM ** REPORT 10 MODIFICATION REQUEST **
-OPEN

-DESC Modify Report 14 to show the current employee address.
 -COM ** REPORT 14 MODIFICATION REQUEST **
-OPEN
-DESC correct report 23 to hold 7 digits in the total fields
-DESC instead of truncating the number.
```

```
 -TYPE PROBLEM
-COM ** REPORT 23 TRUNCATION ERROR **
-OPEN
```

Executing CCFB103 from Another Program

You can execute the utility from another program through the LOAD and CALL macro, as follows:

```
LOAD CCFB103
LR 15,1
CALL (15),(PARM,CRTABL)
```

All parameters are optional. You can omit them from the PARAM list from right to left or by specifying 0. For example:

```
CALL (15),(0,CRTABL)
```

Parameters

The parameter options are as follows:

optaddr

Specifies the address of the options list. The list contains those execution parameters usually specified in the PARM parameter of the EXEC JCL statement.

The option list must begin on a halfword boundary. The first two bytes in the list must contain a binary count of the number of bytes in the remainder of the list. For a null list, the count must be zero.

cridaddr

Specifies the address of an area to hold the eight-byte change request numbers that the execution creates. If one change request is processed, the area can be the minimum eight bytes. If more than one is processed, then each change request ID is placed in the table in the order processed.

Batch Change Request Close

The Batch Change Request Close program (CCFB104) closes change requests. It is available for batch execution, or another program can call it as an interface vehicle. The JCL requires only a program execute statement and SYSIPT statements, as all needed files are dynamically allocated. The LIB/CCF system master file is dynamically allocated based on the SYSMAST keyword of the $CCFGEN macro.

```
// JOB xxxxxxx
//CLOSE JOB,ACCT,PGMR,CLASS=X
// EXEC CCFB104
control statements
      . . .
      . . .
```

Parameters

The parameter options are as follows:

[UNCONDITIONAL|CONDITIONAL]

Specifies whether to continue processing in case of error.

UNCONDITIONAL

(Default) Processes all verbs that can be processed. You can abbreviate this parameter to UNCOND.

CONDITIONAL

Terminates processing if an error occurs. You can abbreviate this parameter to COND.

SYSIPT Control Statements (Required)

The SYSIPT control statements are as follows:

-USER *ccfid*

Specifies the CCF ID of the programmer owning the change request to close, 1 to 8 characters.

-ANAL *analysis*

Specifies the analysis of the problem and solution, 1 to 12 continuous cards. Each analysis record can be up to 64 characters in length.

-AFFT *list*

Specifies the routines affected, 1 to 15 continuous cards. Each AFFT record can be up to 62 characters in length.

-SYSM *date*

System tests modified date.

-SYSV *date*

System tests verified date.

-SYST *list*

System tests added/modified, 1 to 3 continuous cards. Each SYST record can be up to 46 characters in length.

-INPT *list*

Specifies new user input, 1 to 3 continuous cards. Each INPT record can be up to 58 characters in length.

-OUPT *list*

Specifies new user output, 1 to 3 continuous cards. Each OUPT record can be up to 57 characters in length.

-DOCC *list*

Specifies documentation changes, 1 to 3 continuous cards. Each DOCC record can be up to 51 characters in length.

Processing Verb

The processing verb is as follows:

-CLOSE *WOnnnnnn*

Specifies the Change Request number to close.

Optional

Optional values are as follows:

-COM *comment*

Specifies comments.

-EMOD

Marks the end of one change request's control statements.

-END

Marks the end of the entire control stream.

Specifications

You must specify the required control statements as follows:

- You must specify these statements before the first verb statement, in any order.
- Once you specify one of these values, it remains in effect for subsequent verbs until another control statement of the same type changes it.

Example

The following example is a CCFB104 execution.

```
// JOB xxxxxxx
//CLOSE JOB ,pgmr,CLASS=X
// EXEC CCFB104,PARM=COND
-USER PAPERSYS
-ANAL Add section REP100 to report program PAYR023 to
-ANAL generate sub totals for each cost center.

 -AFFT PAYRG03
-AFFT PAYRG05
-SYSM 03/23/03
-SYSV 03/25/03
-INPT None.
 -OUPT Reports 10 and 12 will no show a cost center sub
-OUPT total.

 -DOCC Documentation updated to show new report 10 and
-DOCC 12 format.
 -CLOSE WO001287
-ANAL Change section REP048 in report program PAYR039 to
-ANAL expand the total fields to 7 digits.
 -AFFT PAYRG9

-OUPT Report 23 will now show 7 digit totals.
 -DOCC Documentation updated to show new report 23 format.
 -CLOSE WO001293
/*
/&
```

Executing CCFB104 from Another Program

You can execute the utility from another program through the LOAD or CALL macro, as follows:

```
LOAD CCFB104
LR 15,1
CALL (15),(PARM,CRTABL)
```

All parameters are optional. You can omit them from the PARAM list from right to left, or by specifying 0. For example:

```
CALL (15),(0,CRTABL)
```

Parameters

The parameters are as follows:

optaddr

Specifies the address of the options list. The list contains those execution parameters usually specified in the PARM parameter of the EXEC JCL statement. The option list must begin on a halfword boundary. The first two bytes in the list must contain a binary count of the number of bytes in the remainder of the list. For a null list, the count must be zero.

cridaddr

Specifies the address of an area to hold the eight-byte change request numbers that the execution closes. If one change request is processed, the area can be the minimum eight bytes. If more than one is processed, then each change request ID is placed in the table in the order processed.

LIBDSCAN

The LIBDSCAN utility program compares a production core image library to a production source library to ensure source-to-phase correspondence. LIBDSCAN relies on Source-Load Audit Trail (SLAT) variables to perform the matching. The *Systems Services Guide (VSE/ESA)* describes the installation and use of this program (which you can also use independently of LIB/CCF). LIBDSCAN is often used with the output exit LIBDEXIT or the input exit IEXDSLAT to ensure a standard format for the SLAT variables in CA Librarian members. The *System Services Guide (VSE/ESA)* describes both of these exits.

For more information on the Source-Load Audit Trail Variables, see the entry on the VAR option in the *Batch Command Reference Guide.* See the Source Load Member Verification section in the “LIB/CCF Model System” chapter in this guide for more information of this subject.

LIBDCOPY

LIBDCOPY is a special utility program that copies the current level of an archived member from one master file to another. It determines whether the member exists on the receiving master file and adds or updates the member accordingly. The *Systems Services Guide (VSE/ESA)* contains information on LIBDCOPY.

Chapter 5: LIB/CCF Model System

LIB/CCF is a highly flexible system that you can tailor to a variety of operating requirements. To select an appropriate configuration, site management must review current practices and determine which LIB/CCF system profile options are most useful for managing your application update cycle. To aid in the analysis process, Computer Associates provides sample JCL to allocate a set of libraries that you can use for testing purposes. These libraries are organized in a typical configuration, as described below.

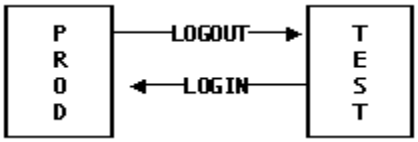
Once you become familiar with the model system provided, you can select the system profile options that are most appropriate for your site.

This section contains the following topics:

Basic Function (see page 117)
System Master File (see page 120)
Model System (see page 122)
Model System Defaults (see page 122)
Model System Requirements (see page 123)
Model System Demo Script (see page 123)
Recommendations for Setting Up Your System (see page 124)
User Exit Points for Customizing LIB/CCF (see page 128)
Suggested Procedure for Emergency Changes (see page 128)
Multiple Programmers Assigned to One Member (see page 129)

Basic Function

The basic function of LIB/CCF is to control the movement of the production CA Librarian member to a test CA Librarian master file where you can change it, then move it back after the change is made. The logout and login functions accomplish this. So, the most basic file requirements for the system are a production CA Librarian master and a test CA Librarian master.

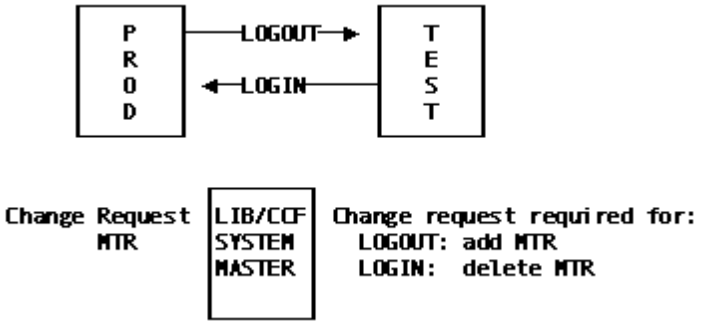


The LIB/CCF system can control up to 65,536 different sets of test and production libraries. One production library can have any number of test libraries associated with it. Any one test library can be associated with any number of production libraries.

Change Request

The LIB/CCF system also provides a control mechanism called a *Change Request*. Each change request for a problem or modification is assigned a change request ID when initiated. All LIB/CCF functions that must be performed to complete this change, such as login and logout, are associated with this change request. Therefore, a LIB/CCF system master file is also required for storing these change requests. Whenever a member is logged out, a Module Tracking Record is created for it.

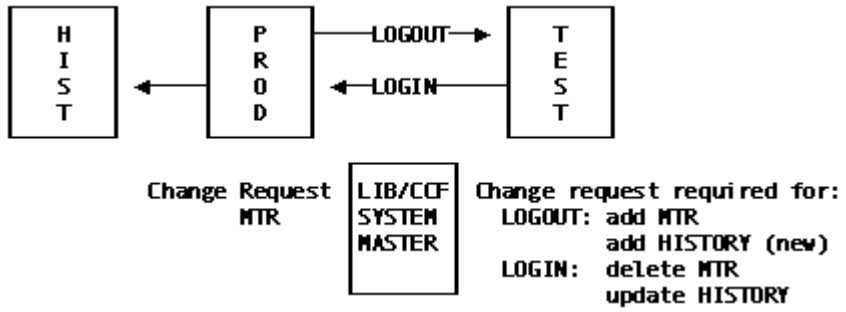
This record exists until the member is logged in. The Module Tracking Record control member is also stored in this LIB/CCF system master file. It contains an entry for every member that was logged out.



History Member

When a member is logged out for the first time, a History member is created. This member contains an abstract of the member, language, description, and the name of the system where it belongs. Each time a login function is performed for that member, this history member is updated to reflect the changes, the number of the change request requesting the changes, and the programmer responsible. Therefore, another required LIB/CCF file is the history master file.

There must be a history master file for each production master file. Several production master files can share one history master file, as long as duplicate member names do not exist.

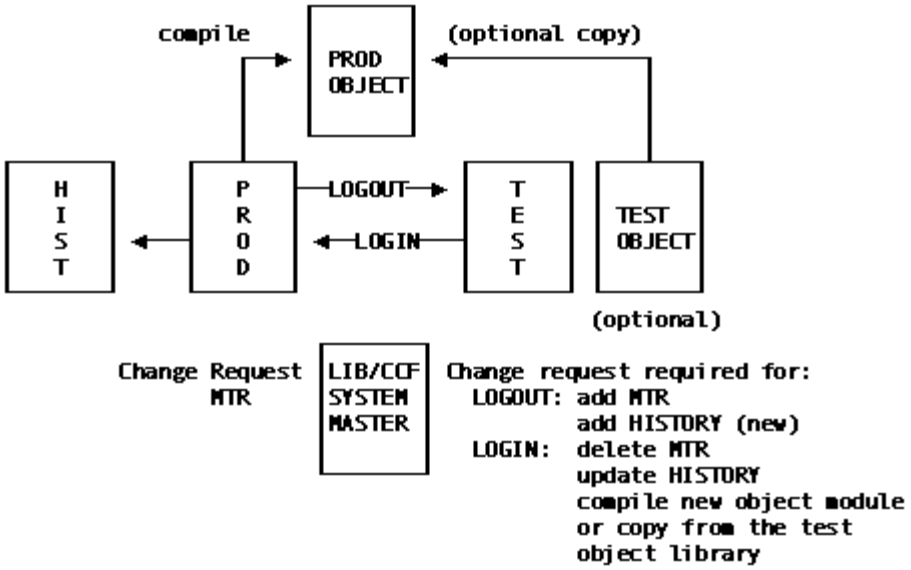


In this model system, the login function compiles the source after the member is moved to production. Therefore, a production object/relocatable library is required for each production master file. Several production master files can share the same object library, as long as there are no duplicate member names.

Test Object

LIB/CCF optionally copies the test object to the production object library instead of recompiling after login. In this case, the system also requires a test object library. Computer Associates does *not* recommend using a test object. If you copy a test object to a production object and then link it, the SLAT variables do not match when LIBRSCAN/LIBDSCAN checks the source master file and load library.

See the Source to Load Member Verification section in the "LIB/CCF Model System" chapter for more information.

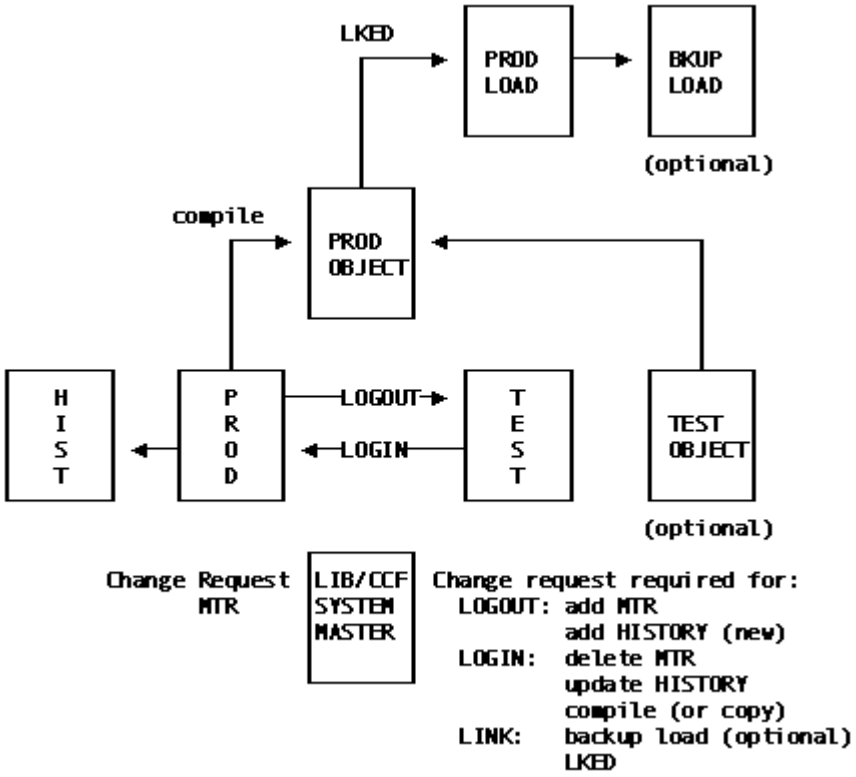


After all source module changes are made and the members are logged in, LIB/CCF provides a link function that link edits the application system. The link-edit requires a member on the production source library containing link-edit input or a complete link-edit job stream.

Link-Edit JCL Member

If the link-edit is performed in batch, the production source library can optionally contain a link-edit JCL member that is submitted as is. The input to the link-edit is the corresponding production object library. The link-edited module is then placed in a production load library, which is the next required LIB/CCF system file.

LIB/CCF provides the additional option of copying the production load module before the link-edit for backup purposes. If this option is exercised, then a backup load library is also required.



Finally, the Change Request is closed. During close processing, the programmer creates a report describing the change. A copy of the report is produced for the user who opened the Change Request.

LIB/CCF can require the control group to control login, logout, and system link functions. If you select any combination of these options, then Module Movement Records (MMR) and System Link Records (SLR) are created to track these processes. These records are also stored on the LIB/CCF system master file.

System Master File

The LIB/CCF system master is an CA Librarian master file. It contains the change requests and a number of control members. The following defines the system master members:

$CCFTnnn

Specifies administrator function tables for ISPF(TSO).

$CCCTnnn

Specifies administrator function tables for ISPF(VM/ESA).

$CCVTnnn

Specifies administrator function tables for LIB/CCF-CA Vollie.

$CHGTnnn

Specifies administrator function tables for LIB/CCF-CA Roscoe.

CCFENQUE

Except LIB/CCF-CA Vollie. Contains enqueue records for resources the LIB/CCF administrators use.

CURWONUM

Contains the current change request number.

JSRCNTL0

Contains Job Submission Records (JSRs).

MMRCNTL0

Contains Module Movement Records (MMRs).

MTRCNTL0

Contains Module Tracking Records (MTRs).

SLRCNTL0

Contains System Link/Bind Records (SLRs).

SLRCNTL1

LIB/CCF-ISPF(VM/ESA) only. Contains additional link-edit information.

S0*nnnnnn*

Contains scheduler information. Applicable only if the Unicenter CA-7 or CA Scheduler interface is used in LIB/CCF. Any change request that has at least one job scheduled has a corresponding SO*nnnnnn* member.

T4*nnnnn*

Except LIB/CCF-CA Vollie. Contains Library Chain members created by the Library Chain Definition Function.

T5*nnnnnn*

Except LIB/CCF-AllFusion CA-Vollie. Contains Library Definition Record (LDR) members created by the Library Chain Definition Function.

WO*nnnnnn*

Specifies an open change request. Each open change request is called by a six-digit identification number prefixed with WO. LIB/CCF assigns these numbers as the change requests are opened, incrementing each request by one.

ZR*nnnnnn*

Specifies a closed change request. The closed change requests are renamed with a prefix of ZR, mainly for performance reasons.

LIB/CCF automatically creates and maintains all of these members.

Model System

To use the model system, you must first install LIB/CCF according to the instructions appropriate for your online environment. See the *Getting Started* for details.

If your site uses both CA Roscoe and ISPF, you can install both LIB/CCF-CA Roscoe and LIB/CCF-ISPF. If you choose to do this, the model systems share all files.

The model LIB/CCF system as installed consists of the following files:

- SYSMAST—The data set name is LIBR.LIBCCF.SYSMAST
- PRODMAST—The data set name is LIBR.LIBCCF.PRODMAST
- TESTMAST—The data set name is LIBR.LIBCCF.TESTMAST
- HISTMAST—The data set name is LIBR.LIBCCF.HISTMAST
- PRODLOAD—The data set names are LIBR.LIBCCF.PRODLOAD (z/OS and OS/390) and LIBRCCF.PROD (VSE/ESA)
- BKUPLOAD—The data set names are LIBR.LIBCCF.BKUPLOAD (z/OS and OS/390) and LIBRCCF.PRODBKUP (VSE/ESA)
- POBJECT—The data set names are LIBR.LIBCCF.POBJECT (z/OS and OS/390) and LIBRCCF.PROD (VSE/ESA)

Note: These data set names are the default names provided at installation time. If you changed the data set names during installation, you must specify the new names in the model system Library Chain Definition (Option 12.11 (LCDF)).

Model System Defaults

The LIB/CCF model system defaults are as follows:

- Programmer performs logout online.
- Control group processes login requests.

- Control group's method of member movement from test to production is batch.
- Control group is responsible for pending batch activity.
- Source module is recompiled to produce the production object module after login to production. Load modules/phases are backed up before relinking a new system.
- Control group processes link requests in batch.

Model System Requirements

Before you can use the LIB/CCF model system, the LIB/CCF administrator must use LIB/CCF Option 12 to update:

- Option 12.0 to specify CNTLKEY(s) for CA Roscoe/Vollie or CNTLIDs for ISPF (control group LIB/CCF keys). All other entries in the System Table are pre-coded to support the model system.
- Option 12.1 to identify AUTHORIZED USERS.
- Option 12.2 to identify PROGRAMMING MANAGERS.
- Option 12.3 to identify PROGRAMMERS.
- If you change the data set names for the model system files during installation, use the Option 12.11 MODIFY CHAIN function to specify the new data set names for chain number 1.

You can define a programmer as reporting to more than one manager. You can also define an individual as belonging to more than one group. One person can be a requestor, a manager, a programmer, and a control group member, too. The LIB/CCF administrator must also review the following JCL skeletons and modify them as necessary to conform to site requirements. For possible changes you should note, see the JCL Skeletons section in the appropriate chapter in this guide for your environment.

Model System Demo Script

After LIB/CCF installation, sample modules are available for use with the demo script. They are TESTMOD (a sample assembler program) and TESTLKED (a link-edit SYSIN stream to link TESTMOD). You can follow the script to demonstrate the system functions. The CA Roscoe key, TSO or VM/ESA user ID, or CA Vollie operator name used for the demo must first be defined as an authorized user, an applications manager, a programmer, and a control group member so that you can demo the system without having to signon as different users.

- As a requestor, open a change request using Option 1.
- As the applications manager, assign the opened change request (to your own CCF key), using Option 2.

- As the programmer, view the assigned request using Option 3. Logout member TESTMOD using Option 4. Supply the change request number and then a LANGUAGE of ASM.
- As the programmer, make a change to the member TESTMOD on the test master file using the interface specific to your environment.
- As the programmer, log in member TESTMOD Using Option 5.
- As a control group member, process the login request using Option 9.
- As the programmer, use Option 6 to link TESTMOD using link input member TESTLKED and using the change request number.
- As a control group member, process the link request using Option 10.
- As the programmer, CLOSE the change request using Option 7.

Recommendations for Setting Up Your System

This section describes our recommendations for setting up your system.

Security Protection

LIB/CCF relies on the management code (MCD) feature of CA Librarian or CA Librarian's external security interface to provide master file security. This feature allows the assignment of a management code (or MCD) to a master file and a security status (either TEST, PROD0, PROD1 or PROD2) to individual members on the master file. The assignment of the MCD to the master file must be done with a batch CA Librarian job for z/OS and OS/390 or VSE/ESA initialized master files, or with the LIB/CMS LIBMAST command for VM/ESA initialized master files. For details on assigning an MCD and the various security statuses, see the *Security Administration Guide.*

Once your LIB/CCF master files are assigned an MCD, you must tell LIB/CCF what the MCD base code for each file is and what the status for members in each file should be. For CA Roscoe, VM/ESA, and TSO, use the Library Chain Definition Facility (or LCDF, Option 12.11). For CA Vollie, use the Production Master File Table (Option 12.6), the History Master File Table (Option 12.7) and the Master File Information Table (Option 12.10). Once the MCD is assigned to the master file and the MCD base code and member status are specified to LIB/CCF, LIB/CCF maintains this information and generates the correct MCD and status whenever they are required for member processing.

Your production source master file should be security protected, with a status of PROD1 or PROD2 for the production members. We also recommend that you have security protection for the test master files. You should specify the LOCK status or security status given to a test member on a test master file when the login function is performed as PROD1 or PROD2. In this way, you lock the test member from further changes when it is logged in. The member on the test master remains in TEST status until it is logged in. The status is changed when the login function is requested.

Source to Load Module Verification

To positively trace the phase or load module to the correct source member version, use the CA Librarian Source-Load Audit Trail (SLAT) variables. These variables refer to member control information such as the member name, programmer name, date and time last updated, level number, and master file data set name. They are entered into the source code (or the compilation file) as literals using the facilities of the particular programming language. CA Librarian dynamically substitutes them at compile or assembly time.

For example, a COBOL program might contain the SLAT variables demonstrated in the following example:

```
01  SLAT-VARIABLE-AREA.
    05 FILLER          PIC X(16) VALUE ‘SLAT VARS START:’.
    05 SLAT-DATEUPD    PIC X(8)  VALUE ‘¢DATEUPD’.
    05 SLAT-TIMEUPD    PIC X(8)  VALUE ‘¢TIMEUPD’.
    05 SLAT-PROGRAMMER PIC X(15) VALUE ‘¢PROGRAMMERNAME’.
    05 SLAT-MODNAME    PIC X(8)  VALUE ‘¢MODNAME’.
    05 SLAT-LEVEL-NO   PIC X(5)  VALUE ‘¢LVNO’.
    05 SLAT-UPDATE-NO  PIC X(5)  VALUE ‘¢UPNO’.
    05 SLAT-DSN        PIC X(44)
       VALUE ‘¢DATA-SET-NAME-FOR-THE-LIBRARIAN-MASTER-FILE’.
    05 FILLER          PIC X(13) VALUE ‘SLAT VARS END’.
```

The variables are substituted with the actual member information at the time of the compile. The substituted values become part of the object module and ultimately the load module or phase. A pre-coded Assembler example is contained in a member named VARIABLE on the System Services file, which can be included in Assembler programs as is. See the *Batch Command Reference Guide* for details on the Source-Load Audit Trail variables

There are two approaches to using the SLAT variables.

- The first approach involves actually updating the source member with the SLAT variables. Once the source member contains the SLAT variables, you can compile the member using the batch CA Librarian (with the -SEL MEMBER,VAR,EXEC control command) or by using the CA Librarian Access Method (LIB/AM or LAM). Both compile techniques substitute the variables with the actual values that they represent. If you update your source members with the SLAT variables, a batch CA Librarian input exit (IEXSLAT for z/OS and OS/390, IEXDSLAT for VSE/ESA) automatically inserts the SLAT variables when batch CA Librarian adds or updates a member.
- The second approach *does not* involve updating the source members with the SLAT variables. Instead, a batch CA Librarian output exit (LIBREXIT for z/OS and OS/390, LIBDEXIT for VSE/ESA) automatically inserts the SLAT variables into the compilation file when you compile a module using the batch CA Librarian (with the -SEL MEMBER,EXEC control command). This method requires that batch CA Librarian perform compiles (rather than LIB/AM).

All of the above mentioned exits (LIBREXIT, LIBDEXIT, IEXSLAT, and IEXDSLAT) provide the SLAT variables in a programming language specific format. For example, for COBOL programs, the exits provide the SLAT variables as VALUE clauses. For Assembler programs, the exits provide the SLAT variables as define constants (DCs). For the exit to determine the programming language that the member is written in, the member must have a three-character CA Librarian language (assigned by the CA Librarian -LANG control statement) that matches a language in the exit's language table.

As distributed, the exits support COBOL, PLI, and Assembler programs and contain a modifiable language table so that you can change existing languages or add new ones. See the *System Services Guide* for details on each exit.

Finally, a program cross checks the load module or phase to the source member and verifies that they match. This program (LIBRSCAN for z/OS and OS/390, LIBDSCAN for VSE/ESA) requires a specific format for the SLAT variables, which the VARIABLE member or the input/output exit programs provide. If your load modules or phases already contain the SLAT variables in a different format, you can use LIBRSCAN/LIBDSCAN by simply modifying the DSECT that maps the format of the SLAT variables. Otherwise, you might find it necessary to modify the program source itself.

The source code for these assembler programs (IEXSLAT, IEXDSLAT, LIBREXIT, LIBDEXIT, LIBRSCAN and LIBDSCAN) are located on the User Contributed Routines (UCR) target library. See the *System Services Guide* for details on installing and using these programs. The following section provides information on incorporating these programs into LIB/CCF.

Incorporating Verification into LIB/CCF

Because LIBREXIT, LIBDEXIT, IEXSLAT, and IEXDSLAT are batch CA Librarian exits, you can easily incorporate them into LIB/CCF as follows:

LIBREXIT

(z/OS and OS/390) Can be invoked during the login compile by adding the parm OEX=LIBREXIT to the batch CA Librarian step of the compile or assembly skeletons.

LIBDEXIT

(VSE/ESA) Can be invoked by the login compile skeleton by executing a version of the batch CA Librarian that was linked with LIBDEXIT.

IEXSLAT

(z/OS and OS/390) Can be invoked at login during the update of the production module using LIBRCOPY. Add the parm IEX=IEXSLAT to the batch CA Librarian step of the LIBRCOPY skeleton.

IEXDSLAT

(VSE/ESA) Can be invoked at login during the update of the production module using LIBDCOPY. The CA Librarian step must execute a version of the batch CA Librarian that was linked with IEXDSLAT.

LIBRSCAN and LIBDSCAN

Stand-alone utilities and, as such, can be executed at any time outside of LIB/CCF.

Considerations for Verification

LIB/CCF assumes that the production compile and link-edit are performed against the production source and object libraries to provide source to load synchronization. Review the following scenarios; they cause a mismatch between the production source and the production executable modules:

- If test object modules are copied to the production object library and then linked, the SLAT variables contain information that pertains to the test master file, *not* the production master file. This results in a discrepancy between the production source and production load.
- If test load modules or phases are copied to the production executable library, the SLAT variables contain information that pertains to the test master file, *not* the production master file. This results in a discrepancy between the production source and production load.

For the time period after a source module is compiled from production and before it is link edited, the source and load do not match. However, this out of synch condition is corrected as soon as the production object module is link edited.

User Exit Points for Customizing LIB/CCF

User data areas are provided in the tracking records LIB/CCF maintains. Sample exit programs are unloaded to your CA Librarian master. For LIB/CCF-CA Roscoe, their names are $CHGU0nn. For LIB/CCF-ISPF(TSO and VM/ESA), their names are $CCFU0nn. For CA Vollie, the names are $CCVU0nn.

You can modify these sample exits or provide like exits if you want to supply data for these record areas. For instance, the sample exits are set up to require specification of a deadline for the implementation of a particular change request.

Other user exit points:

- Verify the data on the logout function panel.
- Provide and verify the data on the HISTORY creation panel.
- Provide a point for creating special reports on login termination.

Suggested Procedure for Emergency Changes

We recommend that your site set up a procedure for emergency changes.

Define an application called EMERGENCY (or use a name of your own choosing). Designate your key operations authority as manager of this application, as the programmer reporting to himself, and define that person's user ID as the emergency key. You can then use the EMERGENCY application for opening and closing emergency change requests to provide a record of such changes.

To designate a user key as the emergency key, perform the following steps:

1. In the System Profile Table (12.0), assign the keyword EMERID (for TSO and VM/ESA) or EMERKEY (for CA Roscoe and AllFusion CA-Vollie) to a user ID.
2. In the User Definition Table (12.1), authorize that user ID to the EMERGENCY application. Define a Chain Group that contains a chain for every production master file that the EMERGENCY application can use. It might suffice to define only one test emergency master file for all of the production master files. However, if duplicate member names exist on the various production master files, an attempt to logout a member that is already logged out into the emergency test master file fails. Additionally, if your site uses QA libraries, you must decide whether you want to define them in the emergency chains.

3. In the Managers Definition Table (12.2), designate the emergency user ID as the manager of the EMERGENCY application with the emergency Chain Group.
4. In the Programmer Definition Table (12.3), designate the emergency user ID as a programmer and as a manager of itself.

You can define more than one emergency application and more than one user ID or key as an emergency key, if necessary. Whoever is defined as having the emergency user ID is automatically granted:

- Multiple logout privileges (regardless of the ALLOWMULT specification in Option 12.0). Multiple logout privileges let the emergency ID log out a member that was already logged out. (See the following section for information of multiple logouts.)
- Control group privileges, which let the emergency ID process login and link-edit requests.

The emergency ID can then open an emergency change request, assign the change request to himself, log out the member, fix it, log it in, do the link, and close the change request.

Multiple Programmers Assigned to One Member

The normal mode of operation for LIB/CCF is to allow only one programmer to log out a member at a time. However, if your site needs multiple programmers to log out and work on the same member at the same time, specify the keyword ALLOWMULT YES in Option 12.0 (PROFILE). ALLOWMULT YES permits and tracks multiple logouts of one member.

With ALLOWMULT YES activated, a member that was logged out for a long-term project, but whose production version needs a quick fix for a bug, can be repaired without undue disruption.

It works this way:

1. Programmer A logs out a member to test master file A.
2. Programmer B logs out the same member to test master file B. The logout displays a panel informing him that programmer A already has the member logged out and provides an option to either continue or terminate the logout. He chooses to continue.
3. Programmer A makes his changes and submits a login request for the member.
4. Programmer B makes his changes and submits a login request for the member.
5. The control group successfully processes the login request from programmer A, which copies the new version back into production.

6. When the control group attempts to process the second request, LIB/CCF issues a message that the member was changed since it was first logged out and presents a panel showing the current production version, the logged out version, and all levels in between.
7. The panel provides the option to generate an Archive Cross Level Report. Select this option. For CA Vollie CCF, it uses the batch CA Librarian -OPT COMPARE function to produce a report showing the differences between the current level and the logged out level of the production member and the differences between the current level and the logged out level of the test member. For CA Roscoe, TSO, and VM/ESA CCF, it uses COMPARATOR II rather than -OPT COMPARE. The compare job contains three steps.
    a. The first step compares the current level in the test or Q/A master file to the level logged out from production.
    b. The second step compares the current level in the production master file to the level logged out from production.
    c. The third step compares the current level of each file.
8. Carefully review the report and make any necessary changes to programmer B's copy, incorporating the changes made by programmer A.
9. Once the changes are reconciled, the control group then attempts the move again and selects the override option on the version check failure panel.

The production member is then updated by programmer B's test member, which now contains all the changes made to the member by programmer A and programmer B.

If you set up your system with ALLOWMULT YES, it is important to heed the following recommendations:

- Have a sufficient number of test master files defined; the member can ONLY be logged out to a test master where it does not already exist.
- For LIB/CCF-CA Vollie, which uses the -OPT COMPARE feature, the test master file must be initialized to support enough archive levels to archive every change that the original programmer must make to the member between the time that he first logs it out and the time that he is ready, all changes reconciled, to log it back in to the production master file. If insufficient archive levels are provided, the archive level that was current during logout is lost as soon as enough levels accumulate on the test master file to exceed the maximum. CA Librarian deletes old levels. Then, if the level that was current when the member was first logged out was deleted, the Cross-Level Report generation job fails and manager or control group intervention is necessary.

- The Production to Test COPY specification in Option 12.4 (Vollie) or Option 12.11 (CA Roscoe, TSO, and VM/ESA) must be UTILITY COPY.

A control group must manage login. When two or more programmers are working on the same program, it is essential that the final authority for updating the production master reside with a control group to enforce proper procedures.

Chapter 6: Exception Handling

Sometimes a problem arises in LIB/CCF requiring the intervention of the LIB/CCF administrator.

This section contains the following topics:

Reassigning a Change Request (see page 133)
Rerunning a Failed Batch Job (see page 133)
Possible Logout Problems (see page 136)
Possible Login Problems (see page 137)
Multiple Administrator Access (z/OS and OS/390 and VM/ESA Only) (see page 142)

Reassigning a Change Request

Managers can reassign change requests from one to another of the programmers who report to them using the REASSIGN command in Option 2. Only the LIB/CCF administrator can use Option 12.9 to reassign a change request from one manager to another.

Rerunning a Failed Batch Job

When LIB/CCF submits a batch job, two important things take place: a Job Submission Record (JSR) is placed in the Pending Job Status queue (Option 11), and a member containing the complete job stream is added to the LIB/CCF system master file. If the job executes successfully, the JSR is automatically removed from the Option 11 queue and the job stream member is deleted from the system master file.

However, if the job fails, the JSR and the job stream member remain for subsequent processing. An examination of the job output shows what steps ran and what steps failed. If there are JCL errors, you must correct the job stream member. To correct the job stream member, you must view the JSR entry for the job in Option 11.

The following sample shows the layout of a JSR, with the job stream member name underlined.

```
              jobname jobnum CR: WOnnnnnn DATE: mm/dd/yy hh.mm.ss PGMR:
              PROD MASTER: data set name
CHAIN: xxxxx  MODULE NAME: moduname TYPE: MOVE Ynnnnnnn
              VERS=mmddhhmm < user data area >
```

For security reasons, the job stream members on the system master file are protected, which means that the MCD or appropriate security level is required to update them. See the *Security Administration Guide* for details on security status and MCD codes. Generally, the LIB/CCF administrator is the person to correct the member (using CA Roscoe, CA Vollie, LIB/CMS, or ELIPS). After correcting the JCL member, the control group or programmer can use the RESUBMIT command from Option 11.

For VSE/ESA

For jobs submitted from VSE/ESA, correct the job stream member before using the RESUBMIT command from Option 11. If the job stream contains an MCD, which is calculated using the date, it can require updating. You must remove job steps that ran before, but that would fail if run again. If the job runs to completion this time with no errors, the JSR is automatically deleted.

For VM/ESA

For jobs submitted from VM/ESA, remove job steps that ran before, but would fail if run again. When necessary, the RESUBMIT command recalculates the MCDs in the job stream. Therefore, do not update the MCDs when resubmitting jobs with the RESUBMIT command. If the job runs to completion this time with no errors, the JSR is automatically deleted.

For z/OS and OS/390

For jobs submitted from z/OS and OS/390, the RESUBMIT command from Option 11 displays the Pending Job Restart panel. Each JCL EXEC statement appears on the Pending Job Restart panel. From this panel, the programmer or the control group can restart the job at a particular step. When necessary, the RESUBMIT command recalculates the MCDs in the job stream. The restart function adds the OS JCL RESTART parameter as the second line of the job card. Therefore, **do not** update the MCDs or add the RESTART parameter to the JCL member when resubmitting z/OS and OS/390 jobs using this service.

```
FRI mm/dd/yy -------- PENDING JOB RESTART ---------- 02.34.42

COMMAND ===>                                    SCROLL ===> PAGE

          Select a STEPNAME to restart, or END to CANCEL
*STEPNAME *----------- J C L R E C O R D ----------------*
 GETTEST //GETTEST  EXEC PGM=LIBRCOPY
 PUTPROD //PUTPROD  EXEC PGM=AFOLIBR,PARM='NRJS',COND=(0,NE)
 DELTEST //DELTEST  EXEC PGM=AFOLIBR,PARM='NRJS',COND=(0,NE)
 STEP1   //STEP1    EXEC PGM=AFOLIBR,PARM='NRJS',COND=(0,NE)
         //STEP2    EXEC ASMHC,PARM='DECK',COND=(0,NE)
CHECKOFF //CHECKOFF EXEC PGM=$CCFB100,COND=(4,LT)
```

The primary commands available from this panel are:

END

Cancels the resubmit.

RESTART ***stepname|stepname.procstepname***

Restarts the job from the specified job step or procedure step. See the *IBM z/OS* and *OS/390 JCL Guide* for a detailed explanation of the RESTART subparameters.

The line command available from the STEPNAME selection list is:

S

Selects a STEPNAME to restart. You must restart a step that executes a procedure with the RESTART command.

Possible Logout Problems

At logout time, an existing member is copied from a production master file to a test master file, either at once by LIB/CCF or later by the control group if logouts are under their control at your site. Normally, the member is copied to the test master file, except in the following cases:

- Member is already on the test master file.
- Member is already logged out through LIB/CCF.
- Control group is in charge of logouts and they reject the request.
- LOGLIMIT is exceeded.
- Logout is undesirable in some way and must be cancelled.

The following sections describe ways to solve these problems as they arise.

Member is Already on the Test Master File

When an existing member is logged out, LIB/CCF is requested to grant the programmer a copy of the member to work on. LIB/CCF places this copy on the test master file specified on the logout panel. But if LIB/CCF finds that the test master file already has a member with the same name, it rejects the logout request, even though the member is not already logged out. The like-named member on the test master file must be examined and deleted, or perhaps renamed, before the logout process can be performed successfully.

Member is Already Logged Out

Generally, when a programmer logs out a member, LIB/CCF does not permit further logouts of that same member. The second programmer to try a logout receives a rejection and an explanatory message. But if LIB/CCF is installed to allow multiple logouts of the same member (ALLOWMULT=YES), LIB/CCF permits them on condition that any multiple logouts must be to different test master files. When a user tries to log out a member that is already outstanding, LIB/CCF displays a list of all the test master files to which the member is logged out. If the test master file that is specified already has the member (someone else is using the same master file), LIB/CCF rejects the request outright. Otherwise, LIB/CCF offers the programmer the option of either continuing or terminating the request.

Control Group Rejects the Logout Request

If the control group rejects a logout request, LIB/CCF marks the login/logout status entry as *REJECTED*. The programmer and the control group decide what the next step should be. In any case, you should delete the login/logout status entry from Option 9 before attempting any further logout of the member.

Member Logout Limit is Exceeded

The LOGLIMIT parameter of Option 12.0 specifies the maximum number of members that a programmer can log out at the same time. When a programmer's attempt to log out a member exceeds the maximum, LIB/CCF rejects the logout. Programmers must then finish work on the other members already logged out and log them in. They can then try the logout again. If this is consistently a problem, consider increasing the LOGLIMIT specification.

Cancelling a Logout

You can cancel unnecessary logouts by locating the entry for the corresponding member in Option 5 and deleting the entry. If the logout request went to the control group but was not acted on, LIB/CCF cancels that request also.

Possible Login Problems

The following conditions can block the login process:

- Member is not on the test master file.
- Program fails in assembly or compilation.
- Multiple logouts are allowed and someone else has logged out the member.
- Control group is in charge of logins and they reject the request.

The following sections describe ways to solve login problems as they arise.

Member is Not on the Test Master File

When a member is logged in, LIB/CCF is requested to update production with the test master file copy of that member. Naturally, if LIB/CCF cannot locate the member on the specified test master file, it cannot perform the login. If the login is for a new member, it might not have been created on the test master file. On the other hand, inadvertent deletion of the member calls for its recovery from the latest backup.

Assembly or Compile Fails

The assembly or compilation step always takes place in batch. For each job that it submits, LIB/CCF places an entry into the Pending Job Status queue, which you can view from Option 11. This entry shows the production and test master file names and the name and number of the batch job.

In case of failure, an examination of the job listing reveals what the errors are. If the job failed because of JCL or non-compilation related errors, you can correct and resubmit the job stream as described in Rerunning a Failed Batch Job earlier in this chapter. But if the error is in compilation and assembly, you must change the member being compiled or assembled. If you already logged the member into production, you must log it out again. First, however, you must process the Pending Job Status queue entry to get rid of the entry. Only then can you log out the member from Option 4.

There are Multiple Logouts

Under certain circumstances, it is possible for a member that is already logged out to be logged out, changed, and logged in again while the first programmer that logged it out is still at work on it. This can happen if an emergency change is necessary to the member or if ALLOWMULT YES was specified in Table 12.0. A programmer attempting to log in a member that is logged out a second time and then logged in again sees the following panel:

```
          MODULE VERSION CHECK FAILURE FOR:  COBMOD

DESTINATION MASTER ===> LIBR.PROD.MAST

LOGGED OUT VERSION ===> 07281055  CURRENT VERSION ===>05251025

GENERATE ARCHIVED CROSS-LEVEL REPORT ===>    OVERRIDE ===>

          ARC    VERSION     PSWD PGMR   DESCRIPTION
CUR VER=>  -0 94/05/25 10:25 QXRF BAKER  COBOL TEST PROGRAM
L/O VER=>  -1 94/07/28 10:55 QXRF BROWN  FIRST INCLUDE
```

Use this panel to request that LIB/CCF generate a Cross-Level Report to help reconcile the differences between the programmer's new version and the version currently on the production master file.

CA Vollie

For CA Vollie LIB/CCF, the Cross Level Compare uses the batch CA Librarian -OPT COMPARE function to produce a report showing the differences between the current level and the logged out level of the production member, and the differences between the current level and the logged out level of the test member.

CA Roscoe

For CA Roscoe and ISPF(TSO) LIB/CCF, the Cross Level Compare uses Comparator II rather than -OPT COMPARE. The compare job contains three steps:

1. Compares the current level in the test or Q/A master file to the level logged out from production.
2. Compares the current level in the production master file to the level logged out from production.
3. Compares the current level of each file.

VM/ESA

For ISPF(VM/ESA) LIB/CCF, the VM/ESA COMPARATOR, which incorporates COMPARATOR II logic, is used. The comparison involves three steps:

1. Compares the current level in the test or Q/A master file to the level logged out from production.
2. Compares the current level in the production master file to the level logged out from production.
3. Compares the current levels of each file.

The comparison report is written to the PRINTER file.

After examining the Cross-Level Reports, the programmer must revise the test member to reconcile all changes and login the member again. LIB/CCF again reports that its version check failed and presents the same panel. As long as the member was not changed after the programmer's first login attempt, he can select OVERRIDE and log in the member. However, if the member was again updated while the test member was being revised, the changes must be reconciled again. This is the case if the number of new levels created since the original logout changed since the attempted login. LIB/CCF provides this information on the panel. If the number changed since the login was postponed for code reconciliation, Cross-Level Reports are generated showing the new changes. You must reconcile those changes again.

If you set up your system with ALLOWMULT=YES, we recommend the following:

- Have a sufficient number of test master files defined. The member can ONLY be logged out to a test master where it does not already exist.
- For LIB/CCF-AllFusion CA-Vollie, which uses the -OPT COMPARE feature, the test master file must be initialized to support enough archive levels to archive every change that the original programmer must make to the member between the time that he first logs it out and the time that he is ready, all changes reconciled, to log it back in to the production master file.
- If insufficient archive levels are provided, the archive level that was current during logout is lost as soon as enough levels accumulate on the test master file to exceed the maximum. CA Librarian deletes old levels. Then, if the level that was current when the member was first logged out was deleted, the Cross-Level Report generation job fails and manager or control group intervention is necessary.
- For LIB/CCF-AllFusion CA-Vollie, the Production to Test COPY specification in Option 12.4 must be U (for Utility Copy). A control group must manage login. When two or more programmers are working on the same program, it is essential that the final authority for updating the production master reside with a control group to enforce proper procedures.

Control Group Rejects the Login Request

If LIB/CCF does not execute login requests immediately but instead channels them to a control group, the control group has the option to reject the login request. If they exercise it, LIB/CCF marks the login/logout status entry as *REJECTED*. Additionally, LIB/CCF puts back on the programmer's login queue (Option 5) an entry for the member marked *REJECTED*. This allows the programmer to log in the member again at a later time.

If you defined a reject library in the Master File Definition Table (AllFusion CA-Vollie) or LCDF (CA Roscoe, TSO, and VM/ESA), then the member is moved to the reject library and deleted from where it currently resides.

The security status of the member on the reject library is defined in the Master File Definition Table (CA Vollie) or LCDF (CA Roscoe, TSO, and VM/ESA). If the STATUS specified is PROD1 or PROD2 and you want to unlock the member to let a programmer make additional modifications, then the control group can use the UNLOCK command in Option 9.

If you are not using reject libraries, then the member remains on the library in its current status. If you want to make the member available to the programmer, use the UNLOCK command in Option 9.

If the programmer deletes the entry for the member from his login queue, the original logout is cancelled (other programmers can now work on the member) and, in effect, cancel all the changes made to the member on the test master file.

The control group should unlock the member using Option 9 before the programmer can delete the Option 5 entry.

The control group or the programmer must delete the login/logout status entry (marked as *REJECTED*) from the queue (Option 9).

Multiple Administrator Access (z/OS and OS/390 and VM/ESA Only)

To ensure data integrity of the system master file tables, LIB/CCF performs enqueue processing on the tables. Any access to one of the administrator suboptions enqueues or locks that option. When the administrator exits the option, the option is dequeued or unlocked.

Because an abnormal termination (that is, a system time-out or abend) can cause an option to remain enqueued erroneously, LIB/CCF provides a manual dequeue mechanism. To demonstrate, review the following scenario:

- Administrator SMITH selects Option 12.5 to review and update information in the table.
- Meanwhile, administrator JONES selects option 12.5 and is presented with the following panel:

```
COMMAND ===>

Option 12.5          may be acquired at this time, however,
     an update attempt by BRANTON will be canceled. Use the
     dequeue command only to cancel an enqueue that remains
     due to a system failure.
```

  As the panel indicates, administrator JONES should “press Enter to return to the previous panel.” However, if JONES suspects that the option is enqueued erroneously (for example, SMITH is out of the office for the day), he can enter the DEQUEUE command. You should only use the DEQUEUE command when you are certain that the other administrator is not updating the option since the command cancels the other administrator’s updates.
- If administrator SMITH is updating Option 12.5 and JONES enters the DEQUEUE command, the following panel displays to SMITH when he tries to save his changes and exit the option:

```
COMMAND ===>

                          << ADMINISTRATOR ENQUEUE >>

The update has be canceled.
```

Note: This enqueue process is performed only on Option 12 suboptions.

Any access to an Option 12 suboption prevents other administrators from accessing it. Remember to exit the option as soon as you are finished.

The Library Chain Definition Function (Option 12.11) shares the same tables among LIB/CCF-CA Roscoe, ISPF (TSO), and ISPF (VM/ESA). All of the other administrator suboptions maintain separate tables for each LIB/CCF system.

Chapter 7: Implementing Quality Assurance Procedures

By implementing quality assurance (Q/A) procedures as part of the change control process, a site can establish the necessary checks and balances in the application development cycle to help ensure the validity of the production source and the production executions. Q/A procedures, whether complicated or simple, facilitate system testing.

This section contains the following topics:

Q/A Process in LIB/CCF (see page 145)
Method One: Defining a Production-Test Promotion Path (see page 147)
Method Two: Defining Q/A Libraries in Promotion Paths (see page 148)
Implementing Method Two in CA Vollie (see page 148)
Verify the Q/A Configuration (see page 163)

Q/A Process in LIB/CCF

A site's Q/A requirements can vary greatly, depending on the number of applications, the number of programmers, and the organization of the data processing department. The LIB/CCF Q/A function lets a site define as many levels of Q/A libraries as needed. Additionally, a site can choose to do quality assurance without introducing another level of libraries into the development cycle.

A basic assumption made in Q/A processing is that the control group is responsible for processing login requests, production movement requests, and all pending batch activity. See the System Profile Table section in the "Administrator Functions in LIB/CCF" chapter for information on defining your system to meet these requirements.

Q/A processing in LIB/CCF is built on the idea that each unique pair of production and test libraries defines a promotion path. A simple path consists of a production master file and a test master file. Members are logged out of the production library into the test master file. Once the changes are complete, the current version of the test member is moved into the original production master file. In between the production and test master file pair, a site can also define Q/A libraries. The order of the Q/A libraries further defines the path that a member takes. If Q/A libraries are present when a programmer initiates the login, a member is moved into the first or lowest level Q/A library in the path. You can also submit a Q/A compile and link at this time. For more information, see the Compile and Link at Login section in the appropriate chapter for your particular LIB/CCF environment.

After testing at the first level of Q/A, the Q/A group can decide to accept or reject the member. If the member is accepted, then a member of the control group uses Option 9 to process the movement request. At this time, if there is another Q/A library defined in the path, then the member is promoted to this next level of Q/A. This process continues until all levels of Q/A are exhausted. Then the production libraries are updated.

If the Q/A group rejects the member and the current library has a reject library associated with it in the library promotion path, the member is moved to this reject library. There, the programmer makes any additional changes. The programmer and management can decide to either correct the problem or to abandon the task entirely. After the control group unlocks the member using Option 9, the programmer merely deletes the entry in Option 5 to make the member available for another logout. If the problem is corrected, the programmer processes the rejected movement request using Option 5 to start the Q/A process over again.

There are basically two methods of implementing Q/A procedures using LIB/CCF.

- Method one uses the locked code on the test master file for Q/A testing and validation. This method is preferred when the site does not have complicated Q/A requirements. This method requires fewer libraries to maintain and less system overhead since additional movements into Q/A libraries are not taking place.
- Method two uses separate libraries for Q/A testing. The libraries must include Q/A source libraries. They can also include reject libraries, and Q/A object (relocatable) and load (core image) libraries. This method is more flexible. It lets a site define more complicated library promotion paths per application or group of applications. However, this method is also more complex to implement. There are more libraries to maintain.

The following sections describe the implementation of each method.

Method One: Defining a Production-Test Promotion Path

The LIB/CCF processing flow for this method follows:

1. The programmer logs members out of a production master file using Option 4. The programmer only sees the Production/Test pairs that refer to the application for which the change request was opened. See the section in this guide that describes the Programming Managers Definition Table Processor and the Master File Definition Table Processor (CA Vollie) or Library Chain Definition Function (all other environments) for more information on relating applications to library pairs through the chain group.
2. The programmer modifies the member on the test master file and performs test compiles and link-edits. The test compiles and links are done using the site's test procedures. They might not be under LIB/CCF control. The programmer performs unit testing at this level.
3. The programmer initiates a login request using Option 5. At this point, the code is protected from further updates because LIB/CCF assigned it a production MCD security status.
4. The Q/A group is notified of the login request by a movement request in Option 9.
5. Before the Q/A group processes the request in Option 9, the Q/A testing is performed. This can include a recompilation and execution and other site-defined tests.
6. The Q/A group accepts or rejects the request. If accepted using Option 9, a member of the Q/A group processes the request. During the processing, the test member is moved from the test master file to the production master file. Depending on site options, the production member can be compiled into the production object library or the test object can be moved into the production object (relocatable) library. If rejected, the Q/A group issues a reject command from Option 9. A member of the Q/A group can then issue the UNLOCK command to make the member available to the programmer for corrections. The programmer is notified of the rejected request by a Rejected Login on the Option 5 display.

To implement this method, first identify the unique production and test library pairs. Then define the library pairs to LIB/CCF using the Master File Definition Table Processor (CA Vollie) or Library Chain Definition Function (all other environments). For this method of Q/A, the important field is the LOCK field.

Lock specifies the security status to assign to members after the programmer initiates a login request. This is the mechanism that freezes the code from further updates once the programmer is satisfied with the changes. This entry is essential to this method of implementing Q/A processing. Specifying PROD1 allows the programmer to view the members but denies update access. To prevent the programmer from both viewing and updating the member during Q/A testing, specify PROD2. See the *Security Administration Guide* for more information about this facility.

Method Two: Defining Q/A Libraries in Promotion Paths

The LIB/CCF processing flow for this method follows:

1. The programmer logs members out of a production master file using Option 4. The programmer sees only the Production - Test pairs that refer to the application for which the change request was opened. See the Manager Definition Table Processor and the Master File Definition Table Processor (CA Vollie) or Library Chain Definition Function (all other environments) sections in the “Administrator Functions in LIB/CCF” chapter for more information on relating applications with library pairs through the Chain Group.
2. The programmer modifies the member on the test master file and performs test compiles and link-edits. The test compiles and links are done using the site’s test procedures. They might not be under LIB/CCF control. The programmer performs unit testing at this level.
3. The programmer initiates a login request using Option 5. At this point, the member on the test master file is copied to the first or lowest level Q/A library defined in the library promotion path. Additionally, you can define your system to perform a Q/A compile and link. You can also define the system to copy the test object module to a Q/A object (relocatable) library.
4. A movement request in Option 9 notifies the Q/A group of the login request.
5. The Q/A group performs the first or only level of testing.
6. The Q/A group accepts or rejects the request. If accepted, the Q/A group processes the movement request using Option 9. This processing includes moving the member to the next Q/A library in the promotion path or to the production file if there are no more Q/A libraries. Again, the compile and link or the move into object can also occur at this time. If rejected, the Q/A group issues a reject command from Option 9. If a reject library was related to the Q/A library, then the member is moved to this reject library. The reject library can be the same as the test library. The programmer is notified of the rejection by a Rejected Login on the Option 5 display. The site defines the MCD security status assigned to a member when it was moved into the Q/A library and the reject library in the Master File Definition Table (CA Vollie) or the Library Chain Definition Function (all other environments).

To implement this method, see the Library Chain Definition Function section. If you are using LIB/CCF-CA Vollie, refer to the steps in the following sections of this chapter.

Implementing Method Two in CA Vollie

For LIB/CCF-CA Vollie, the following sections detail the necessary steps to implement and verify a Q/A configuration.

Step 1: Define the Library Promotion Paths

Identify for each unique production-test library pair:

- Name of the test master file
- Name of the production master file
- Names of the Q/A libraries, in order of promotion, from the first level after test to the last level just before production
- Names of the reject libraries to relate to each Q/A library.

Note: The only required libraries are production and test. You can define any number of Q/A levels. Reject master files can be the same as the test master file.

You should also decide what level of Q/A you want to enter after a rejected request is reprocessed. By default, the processing of a rejected login moves the member to the first level of QA. You can decide to move rejected requests to another Q/A library. You must also define the name of the Q/A library used in this case.

Step 2: Complete the Q/A Worksheet

A sample Quality Assurance Worksheet is provided in the member QASHEET on the System Services File. You can print this member using the batch CA Librarian Utility -PRINT facility.

```
      Q U A L I  T Y  A S S U R A N C E  W O R K S H E E T

      Chain Group : AP0000000000

         | Entry  | File-ID        | Status | Copy | Compile/Move
         | Type   |                |        | All/ | For Object
         |        |                |        | Curr |
----------+--------+----------------+--------+------+--------------
Production|   P    |                |        |      |
          |        |                |        |      |
          |        |                |        |      |
----------+--------+----------------+--------+------+--------------
          |        |                |        |      |
          |        |                |        |      |
          |        |                |        |      |
----------+--------+----------------+--------+------+--------------
          |        |                |        |      |
          |        |                |        |      |
          |        |                |        |      |
----------+--------+----------------+--------+------+--------------
 QA3      |   Q    |                |        |      |
          |        |                |        |      |
          |        |                |        |      |
----------+--------+----------------+--------+------+--------------
 Related  |   R    |                |        |      |
 Reject   |        |                |        |      |
          |        |                |        |      |
----------+--------+----------------+--------+------+--------------
 QA2      |   Q    |                |        |      |
          |        |                |        |      |
          |        |                |        |      |
----------+--------+----------------+--------+------+--------------
 Related  |        |                |        |      |
 Reject   |        |                |        |      |
          |        |                |        |      |
----------+--------+----------------+--------+------+--------------
 QA1      |   F    |                |        |      |
          |        |                |        |      |
          |        |                |        |      |
----------+--------+----------------+--------+------+--------------
 Related  |   R    |                |        |      |
 Reject   |        |                |        |      |
          |        |                |        |      |
----------+--------+----------------+--------+------+--------------
 TEST     |   T    |                |        |      |
          |        |                |        |      |
          |        |                |        |      |
----------+--------+----------------+--------+------+--------------
```

The following describes the fields that appear on the Worksheet:

Chain Group

Enter the chain group to which this library promotion path is related. The chain group is defined in the Programming Managers Definition Table.

Entry Type

Do not enter anything in this column. The entry type is defined in the Master File Definition Table. This information is used in the next step.

File-id

Enter the file ID of the master file as defined in step 1.

Status

Enter the MCD security status LIB/CCF should use when adding members to this master file.

Copy All/Curr

A site chooses which copy utility to use when copying members to the master file specified. A site can choose to use only the current level of the sending member to update the master file. Enter *CURR* to select this option. A site can also choose to completely replace the member on the master file with all the levels of the sending member. Enter *ALL* to select this option. If movement is done online, ignore this field.

Compile/Move For Object

When the source member is moved into this library, you can also recompile it. Alternatively, you can copy the test relocatable module from the relocatable library associated with the sending master to the relocatable library associated with the target master. Enter *MOVE* to move the object along with the source. Enter *COMPILE* to recompile the source into the related relocatable library. To do neither, leave the entry blank.

Complete a work sheet for each unique production-test library pair in a chain group. After you enter all of the library promotion pairs in the work sheet, move on to step 3. You can use the blank rows after production if you are defining more than three levels of Q/A.

```
      Q U A L I  T Y  A S S U R A N C E  W O R K S H E E T

      Chain Group : AP0000000000

         | Entry  | File-ID        | Status | Copy | Compile/Move
         | Type   |                |        | All/ | For Object
         |        |                |        | Curr |
---------+--------+----------------+--------+------+--------------
Production|   P    | ap.prod        | prod1  | curr | compile
         |        |                |        |      |
         |        |                |        |      |
---------+--------+----------------+--------+------+--------------
         |        |                |        |      |
         |        |                |        |      |
         |        |                |        |      |
---------+--------+----------------+--------+------+--------------
         |        |                |        |      |
         |        |                |        |      |
         |        |                |        |      |
---------+--------+----------------+--------+------+--------------
 QA3     |   Q    | n/a            | n/a    | n/a  | n/a
         |        |                |        |      |
         |        |                |        |      |
---------+--------+----------------+--------+------+--------------
 Related |   R    | n/a            | n/a    | n/a  | n/a
 Reject  |        |                |        |      |
         |        |                |        |      |
---------+--------+----------------+--------+------+--------------
 QA2     |   Q    | ap.qa2         | prod1  | all  | move
         |        |                |        |      |
         |        |                |        |      |
---------+--------+----------------+--------+------+--------------
 Related |   R    | my.test        | test   | all  | move
 Reject  |        |                |        |      |
         |        |                |        |      |
---------+--------+----------------+--------+------+--------------
 QA1     |   F    | ap.qa2         | test   | all  | move
         |        |                |        |      |
         |        |                |        |      |
---------+--------+----------------+--------+------+--------------
 Related |   R    | my.test        | test   | all  | move
 Reject  |        |                |        |      |
         |        |                |        |      |
---------+--------+----------------+--------+------+--------------
 TEST    |   T    | my.test        | test   | all  | n/a
         |        |                |        |      |
         |        |                |        |      |
---------+--------+----------------+--------+------+--------------
```

The above sample shows a completed Q/A worksheet for a library promotion path. The worksheet indicates that members are logged out of AP.PROD to MY.TEST. After the programmer makes the modifications, the test source module is moved to AP.QA1. The test object is moved to the relocatable library related to AP.QA1. If the QA group accepts the request, the source member from AP.QA1 and its related object are moved to AP.QA2 and its related relocatable respectively. On acceptance of this request, only the current level of the source module on AP.QA2 updates the production source module on AP.PROD. The source module on AP.PROD is compiled into the relocatable library related to the production master file.

If a request is rejected, the member is moved back to MY.TEST where corrections can be made. The members are in the unprotected TEST status when copied into MY.TEST when the member is moved from production or a Q/A library. However, as members are promoted up the library path, they are placed in a read-only status, PROD1.

Step 3: Interpret the Q/A Worksheet

Interpreting the worksheet defines the necessary Master File Definition Table entries. To facilitate this process, the following reviews the valid entry types for this table.

P type entry

Relates a production master file to a test master file. A production/test master file pair must be unique. That is, a production master file can be related to more than one test master file, and a test master file can be related to more than one production master file. However, each discrete production and test pair must have only one P entry. Members are moved from the Production Master File to the Test Master File. If there are no Q or F type entries for the library promotion pair, then this entry also defines that members are moved to the Production Master File from the Test Master File.

Q type entry

Relates a Q/A library to a lower level Q/A library, a Q/A library to a reject library, and a last or final Q/A library to a production master file. A lower level Q/A library can only appear in one Q type entry.

Members are moved from the lower level Q/A library to the higher level Q/A library, from the final Q/A library to the production library, or from the reject library to the Q/A library. A Q/A library can only appear as a lower level library in one Q type entry per chain group.

For example, the following entries are invalid because AP.QA1 appears as a From Q/A Master File in more than one entry in the same chain group.

```
Entry Number:   002
Entry Type: Q               Chain group AP......
 To Q/A Master File
AP.QA2...........
 From Q/A Master File
AP.QA1...........

Entry Number:   003
Entry Type: Q               Chain group AP......
 To Q/A Master File
AP.QA3...........
 From Q/A Master File
AP.QA1...........
```

By default, a reprocessed rejected login request moves the member from the reject library to the first QA master file in the library promotion path. A site can decide to use a different QA library when a rejected request is reprocessed. If this is the case, then a Q entry must exist in the Master File Definition Table that defines this relationship.

```
Entry Number:   008
Entry Type: Q               Chain group AP......
 To QA Master File
AP.QA2...........
 From QA Master File
AP.REJECT........
```

F type entry

Relates the first QA library in a promotion path to the production master file. Members are moved from the test master file defined in the P entry, to the first QA library specified in the F entry. The F entry allows a site to use the same test master file in more than one library promotion path per chain group. For example, if a site defined the following two paths for chain group AP:

```
AP.PROD   AP.PROD1
AP.QA2A   AP.QA2B
AP.QA1A   AP.QA1B
MY.TEST   MY.TEST
```

The two F type entries:

```
Entry Number:   004
Entry Type: F               Chain group AP......
 To QA Master File
AP.QA1A.........
 For Production Master File
AP.PROD..........

Entry Number:   005
Entry Type: F               Chain group AP......
 To QA Master File
AP.QA1B.........
 For Production Master File
AP.PROD1.........
```

A production master file can only appear in one F type entry per chain group.

R type entry

Relates a QA library to a reject library. Members are moved from the QA library to the reject library. There can be more than one reject library related to a single QA library. When the QA group issues the reject command, the R type entries related to the QA library where the member resides at the time of rejection displays. The QA member then chooses one of the relationships. Further, a single reject library can appear in more than one R type entry in the same chain group. For example, the following R type entries are valid.

```
Entry Number:   005
Entry Type: R               Chain group AP......
 From QA Master File
AP.QA1...........
 To Reject Master File
AP.REJECT.........

Entry Number:   006
Entry Type: R               Chain group AP......
 From QA Master File
AP.QA1...........
 To Reject Master File
MY.TEST...........

Entry Number:   007
Entry Type: R               Chain group AP......
 From QA Master File
AP.QA2...........
 To Reject Master File
AP.REJECT.........
```

To interpret the worksheet:

- Match the P and T entry types from the worksheet to form P type entries in the Master File Definition Table.
- Match the P and first Q entry types from the worksheet to form F type entries in the Master File Definition Table.
- Match each Q and R entry types from the worksheet to form R type entries in the Master File Definition Table.
- Match the last Q entry type to the P entry type from the worksheet to form Q type entries in the Master File Definition table.
- Match the lower level Q entry types to the higher level Q entry types from the worksheet to form Q type entries in the Master File Definition Table until the last or final Q entry type from the worksheet is defined as a To QA Master File in the Master File Definition Table.

From the previous sample worksheet, the following Master File Definition Table entry types are identified:

'P'	'F'	'Q'	'R'
AP.PROD	AP.QA1	AP.PROD	AP.QA1
MY.TEST	AP.PROD	AP.QA2	MY.TEST
		AP.QA2	AP.QA2
		AP.QA1	MY.TEST

Step 4: Make the Appropriate Table Entries

Using information gathered on the worksheet, enter the P, Q, F, and R type entries identified in Step 3 above. For our example, the fields of interest in the Master File Definition Table follow.

Entry Type

P, Q, R, or F, as identified from the work sheet.

Production Master File-id

Specify the file ID from the work sheet used in P and F type entries.

Status

Specify the status from the work sheet.

Copy

If ALL is specified on the worksheet, enter U (for Utility). If CURR is specified on the worksheet, enter S (for special). CA Librarian provides two copy utilities, the utility Copy and a special copy called LIBDCOPY. The site chooses which of these utilities to use when copying members to this file if movement is in batch. This field is ignored if movement is online.

Test Master File-id

Specify the file ID from the work sheet.

To Q/A Master File-id

This field displays for Q and F types. For each Q type entry, specify the name of the higher level QA library. For the final Q entry, specify the production master file. If a reprocessed rejected request is promoted to a QA library other than the first QA library, specify the name here. For F type entries, specify the name of the first QA library.

From Q/A Master File

This field displays for Q and R type entries. For Q types, specify the file ID of the lower level QA library. For R types, specify the file ID of the QA library where the reject library is related.

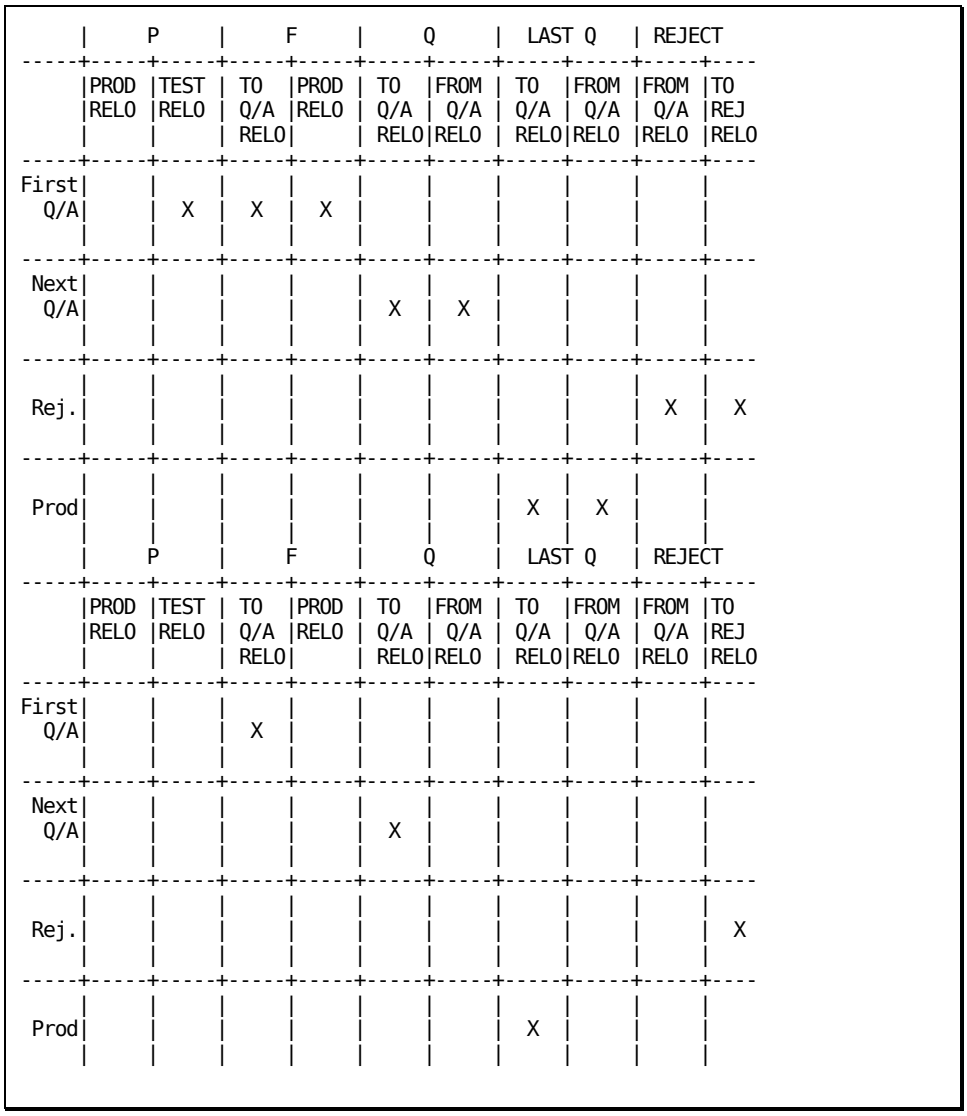
To Reject Master File

This field displays only for R types. Specify the file ID of the reject library where rejected members are copied. For Production Master File, specify the file ID of the production master file from the worksheet.

For more information, see the Master Definition Table Processor section in Chapter 3, “Administrator Functions in LIB/CCF.”

You must also enter the related relocatable libraries. The following describes the rules for these table entries.

- To move the test object module into a relocatable library related to the first QA library, enter a Related Test Object in the P entry. Also enter the Related To Q/A Object and the Related From Object File in the F entry.
- To compile into the relocatable library related to the first QA library, enter only a Related To Q/A Object File in the F entry.
- To move the object module from a lower level QA file to a higher level QA file, enter both a Related To Q/A Object File and a Related From Q/A Object File in the Q entry that defines the lower level QA library and the higher level QA library.
- To compile the source into a relocatable library and copy it into a higher QA library, enter only a Related To Q/A Object File in the Q entry that defines the lower level library and the higher level library.
- To move the object module from a final QA relocatable library into the production relocatable library, enter a Related To Q/A Object File and a Related To Q/A From Object File in the Q entry that relates the final QA library to the production library.
- To compile into the production relocatable library, specify only a Related To Q/A Object File in the Q entry that defines the final QA library to the production file.
- To move the object module associated with a QA source module into a relocatable library related to a reject library, enter a Related From Q/A Object File and a Related To Reject Object File for the R entry that defines the QA library to the reject library.
- To compile a QA source module into a related reject relocatable library, enter only a Related To Reject Object File for the R entry that defines the QA library to the reject library, illustrated by the chart that follows:



	P		F		Q		LAST Q		REJECT	
	PROD RELO	TEST RELO	TO Q/A RELO	PROD RELO	TO Q/A RELO	FROM Q/A RELO	TO Q/A RELO	FROM Q/A RELO	FROM Q/A RELO	TO REJ RELO
First Q/A		X	X	X						
Next Q/A					X	X				
Rej.									X	X
Prod							X	X		

	P		F		Q		LAST Q		REJECT	
	PROD RELO	TEST RELO	TO Q/A RELO	PROD RELO	TO Q/A RELO	FROM Q/A RELO	TO Q/A RELO	FROM Q/A RELO	FROM Q/A RELO	TO REJ RELO
First Q/A			X							
Next Q/A					X					
Rej.										X
Prod							X			

Using the sample worksheet, the following Master File Table entries are made.

```
Entry Number:   001
Entry Type: P                   Chain group AP......
 Production Master File-id          Status Copy
AP.PROD.........                   PROD1 S
Test Master File-id                Status Copy
MY.TEST.........                   TEST U
Related Production Object File
...................
 Related Test Object File
MY.OBJECT.........

Entry Number:   002
Entry Type: F                   Chain group AP......
 To Q/A Master File-id              Status Copy
AP.QA1..........                   PROD1 U
From Production Master File-id     Status Copy
AP.PROD.........                   PROD1 S
Related To Q/A Object File
APQA1.OBJECT......
 Related From Production Object File
MY.TEST...........

Entry Number:   003
Entry Type: Q                   Chain group AP......
 To Q/A Master File-id              Status Copy
AP.PROD.........                   PROD1 S
From Q/A Master File-id            Status Copy
AP.QA2..........                   PROD1 U
Related To Q/A Object File
APPROD.OBJECT.....
 Related From Q/A Object File
...................

Entry Number:   004
Entry Type: Q                   Chain group AP......
 To Q/A Master File                 Status Copy
AP.QA2..........                   PROD1 U
From Q/A Master File               Status Copy
AP.QA1..........                   PROD1 U
Related To Q/A Object File
APQA2.OBJECT......
 Related From Q/A Object File
APQA1.OBJECT......

Entry Number:   005
Entry Type: R                   Chain group AP......
 From Q/A Master File-id            Status Copy
AP.QA1..........                   PROD1 U
To Reject Master File-id           Status Copy
```

```
MY.TEST...........                TEST U
Related From Q/A Object File
APQA1.OBJECT......
 Related To Reject Object File
MY.OBJECT..........

Entry Number:   006
Entry Type: R                   Chain group AP......
 From Q/A Master File-id            Status Copy
AP.QA2............                PROD1 U
To Reject Master File-id           Status Copy
MY.TEST...........                TEST U
Related From Q/A Object File
APQA2.OBJECT......
 Related To Reject Object File
MY.OBJECT..........
```

Production Master File definitions and History Master File definitions are required only for P type entries in the Master File Definition Table.

Verify the Q/A Configuration

Once the table entries are complete, you should perform a trial cycle to verify the results.

1. Open a change request for the application associated with the chain group (Option 1).
2. Assign the request to a programmer (Option 2).
3. Log a member out of the production master file (Option 4). Verify that only the P type entries for the chain group are displayed.
4. Initiate a login (Option 5). Verify that the source module was moved to the first QA library and that LIB/CCF performed the appropriate action for the object module (that is, compiled, moved, or neither). Also check that the target source module was updated according to your specification. That is, make sure that the target source module was completely replaced by the sending source module, or that the target source module was updated with only the current level of the sending module.
5. Process the movement request (Option 9). Verify that the member is now on the second level QA library and that LIB/CCF performed the appropriate action for the object module. Also check that the target source module was updated according to your specification. That is, the sending source module completely replaced the target source module or that the target source module was updated with only the current level of the sending module. Repeat this step until the final QA library is reached.

Process the final movement request into production (Option 9). Verify that the production source file now contains the final QA source module and that LIB/CCF performed the appropriate action for the object module. Also check that the target source module was updated according to your specification. That is, the sending source member completely replaced the target source module or that the target source module was updated with only the current level of the sending member.

Chapter 8: Scheduling z/OS and OS/390 Batch Jobs in LIB/CCF

For CA Roscoe and ISPF(TSO) sites, you can configure LIB/CCF to take advantage of either CA 7 or CA Scheduler letting you schedule an application for promotion to the production environment at a specified time. Not only does this assist in coordinating the movement of large application systems into production, it lets you pre-stage and schedule this process for off peak hours. An additional feature lets you establish predecessor relationships for scheduled jobs.

Based on these relationships, CA 7 or CA Scheduler monitors all scheduled jobs, assuring that all predecessor jobs have successfully executed before submitting a job. The programmer or the control group can schedule jobs, depending on how LIB/CCF is configured.

This section contains the following topics:

Implementation Considerations (see page 166)
Operation (see page 169)

Implementation Considerations

There are four Option 12.0 (PROFILE) keywords that pertain to job scheduling:

- CA7JCLID
- CA7OPERID
- CA7PASS
- SCHEDULER

If you are using Unicenter CA-Scheduler, only the SCHEDULER keyword is necessary. If you are using Unicenter CA-7, the SCHEDULER keyword and some combination of the other keywords are required. See the “Administrator Functions in LIB/CCF” chapter for details on each keyword.

LIB/CCF provides a choice of online or batch processing for activities such as movement to and from production and link-edits. Since both scheduling products schedule batch jobs, several Option 12.0 keywords affect which activities are actually eligible for job scheduling. These keywords are:

- CLINK
- CMOVEMENT
- LKED
- PMOVEMENT
- TMOVEMENT

For example, if you configure LIB/CCF to let the control group perform online link-edits by specifying LKED CNTL and CLINK ONLINE in Option 12.0, the control group cannot schedule link-edits as they are done online. For the control group to schedule link-edits, you need to specify CLINK BATCH in Option 12.0. See the “Administrator Functions in LIB/CCF” chapter for details on each of these keywords.

Since scheduled jobs must have unique job names, LIB/CCF generates a job name of Y*nnnnnnn*, where *nnnnnnn* is a random number. This job name overrides the job name specified on the job card in Option 0.

Option 12.5 (LANGUAGE) correlates JCL skeletons with a language. When a member is logged out for the first time, the programmer assigns a language to it. At login, LIB/CCF uses this language to determine which JCL skeleton is submitted to compile the member.

You can also assign each language a priority, which orders the members in the Group Processing selection list, available from Options 5 and 9. This is useful when the order in which members are processed is important. You can use this default order for job scheduling, or you can further refine it at scheduling time. So, before implementing a scheduler, carefully review Option 12.5 for optimal prioritization of languages. See the Language Definition Table (Option 12.5) section in the "Administrator Functions in LIB/CCF" chapter for more information.

For CA 7

If you specify CA 7 as the scheduler to use, you need to install the CA 7/CA Librarian interface. See the *CA 7 Job Management Installation Reference Guide* for details on link editing the interface program.

Additionally, you must add the following DD statement to the CA 7 online JCL for each LIB/CCF system master file.

```
//JCLnnn DD DISP=SHR,DSN=your.libccf.sysmast.name
```

The value of *nnn* must match the CA7JCLID keyword specified in Option 12.0.

Also add the following JCL statement to the CA 7 Initialization Deck for each LIB/CCF system master file.

```
JCL DSN=your.libccf.sysmast.name,INDEX=nnn,DSORG=LIB,MCD=base
```

The value of *nnn* of the INDEX parameter of the JCL statement must be the same value as *nnn* on the DD statement and the CA7JCLID keyword in Option 12.0. The base value of the MCD parameter must be the MCD *base code* for the LIB/CCF system master file.

Also add the following APPLCTN cards to the Initialization Deck.

```
APPLCTN,NAME=SASSLIBR,ATTR=PERM
APPLCTN,NAME=SASSPROG,ATTR=LOAD
```

Place the APPLCTN statement for SASSLIBR ahead of the APPLCTN statement for SASSPROG, as shown.

The CA 7 operator IDs must match the CA7OPERID keywords specified in Option 12.0.

The CA 7 logon password, if one is used, must match the CA7PASS keyword specified in Option 12.0.

For CA Scheduler

If you specify CA Scheduler as the scheduler to use, the operator command interface program CAJCCMD0 is used. You must specify SCHDCMD=NOCTL as a generation option in the CAIJGEN macro. This option executes CAJCCMD0 from a non-scheduler controlled job. See the *Unicenter* CA-Scheduler Job *Management System Programmers Guide* for details.

Note: This is no longer valid for CA Scheduler release 7.2. See the *CA Scheduler Job Management System Programmers Guide* for details.

Operation

If you specified the SCHEDULER keyword in Option 12.0, you can invoke job scheduling from the following LIB/CCF functions:

- If Option 12.0 specifies PMOVEMENT BATCH or PMOVEMENT ONLINE, the programmer can schedule logins from the Group Processing selection list from Option 5. If you specify PMOVEMENT BATCH, the entire login process is performed in batch. However, if you specify PMOVEMENT ONLINE, only the compile is performed in batch, so the compile job can be scheduled. Once you enter the PROCESS command from the list, the Job Scheduling Information panel displays.
- If Option 12.0 specifies LKED BATCH, the programmer can schedule link-edits from Option 6. Once the programmer provides the required link-edit information, the Job Scheduling Information panel displays.
- If Option 12.0 specifies either:
  - TMOVEMENT CNTL and CMOVEMENT BATCH
  - PMOVEMENT CNTL and CMOVEMENT BATCH
  - PMOVEMENT CNTL and CMOVEMENT ONLINE

  The control group can schedule a variety of jobs from the Group Processing selection list from Option 9. The actual type of job depends on the selection list command entered. When you enter either the COMPILE, LOGOUT, MOVE, PROCESS, or REJECT command from the list, the Job Scheduling Information panel displays.
- If Option 12.0 specifies LKED CNTL and CLINK BATCH, the control group can schedule link-edits from Option 10. Once the you enter PROCESS command for a System Link/Bind Request, the Job Scheduling Information panel displays.
- Depending on the Option 12.0 JSQUEUE specification, either the programmer or the control group can view the status of jobs already scheduled from LIB/CCF from Option 11. Once you enter the SSTATUS command from Option 11, the Scheduler Status panel displays.

  Additionally, if Unicenter CA-7 is specified as the scheduler, jobs can be redemanded (rescheduled) from Option 11.

See the *LIB/CCF User Guide* for details on using the scheduler interface.

Chapter 9: DB2 for z/OS and OS/390 Support in LIB/CCF

DB2 for z/OS and OS/390 is an IBM database management system. Access to and manipulation of data DB2 for z/OS and OS/390 manages is accomplished through SQL. You can imbed SQL in procedural language programs (that is, COBOL, PL1, or assembler programs).

All programs accessing DB2 for z/OS and OS/390 are precompiled. The precompiler produces a modified source program that is passed to the compiler/assembler and a database request module (DBRM) containing information about the required DB2 for z/OS and OS/390 access. One DBRM is created for each program requesting DB2 for z/OS and OS/390 services. DBRMs are stored in a PDS. They are used as input to the bind process. The DB2 for z/OS and OS/390 bind, analogous to a link-edit, validates access to the data, verifies authorization, selects an access path, and builds a control structure called an application plan or package (for DB2 release 2.3 and above). The bind for a plan must be completed before a DB2 for z/OS and OS/390 application can execute.

You can configure LIB/CCF to provide change control functions specific to the DB2 for z/OS and OS/390 environment. The precompile process occurs under the control of LIB/CCF as part of the login function. For programs that are self-contained, LIB/CCF manages the precompile, compile, bind, and link processes with a single request as part of the login function. To create plans that require more than one DBRM, the bind request can be initiated in a separate request through an extension of the system link request option. The Library Chain Definition Facility (LCDF) supports the addition of an operating system library for DBRM management. Optionally, LIB/CCF maintains a table of information describing the DB2 for z/OS and OS/390 programs being managed.

Note: CCF Option 0 (CCF PARMS) allows for the specification of only one DB2 for z/OS and OS/390 subsystem. It is possible to accomodate multiple DB2 for z/OS and OS/390 subsystems at a site. This can be done by creating a separate skeleton for each language and DB2 for z/OS and OS/390 subsystem, and hard coding the subsystem in each flavor of the skeleton.

Reports are available to describe:

- Application plans, including the names of the libraries where the source members and related DBRMs are found
- Number of the change request that resulted in the current production version
- ID of the person who created the plan or package.

See the *LIB/CCF User Guide* for sample DB2 for z/OS and OS/390 reports.

This section contains the following topics:

Implementation Considerations (see page 173)
DB2 for z/OS and OS/390 LIB/CCF Audit Trail (see page 175)

Implementation Considerations

LIB/CCF manages the update cycle for DB2 for z/OS and OS/390 source programs and the associated bind required to create an application plan or package. You can store bind parameters in a production master file that LIB/CCF manages. Bind parameters are keywords and values that specify DBRM names, DBRM library names, and package list entries. They are referred to in this documentation as the *bind* options *members.* The implementation of DB2 for z/OS and OS/390 support affects several areas or functions of LIB/CCF, listed following:

- DB2 for z/OS and OS/390 support is available in LIB/CCF CA Roscoe, LIB/CCF TSO, and LIB/CCF VM/ESA (for batch jobs submitted to an z/OS and OS/390 machine).
- Batch and online bind are available from LIB/CCF CA Roscoe and LIB/CCF TSO. LIB/CCF VM/ESA does not support an online bind, but does support batch bind.
- Bind processing is an extension of Option 6 LINK processing. Therefore, the Option 12.0 (PROFILE) keywords that control LINK processing (CLINK, LKED, LNKNTEGRTY, and LDISPLAY) also apply to bind processing. For example, if you specify LKED BATCH, batch jobs perform both link-edits and binds.
- From Option 5 (LOGIN), the BIND command lets the programmer:
  - Modify an existing, logged out bind options member, or
  - Create a new bind options member through the DB2 for z/OS and OS/390 PLAN UPDATE panel.
- Because the bind options members are input to the bind, you probably want the login process for these members to simply move them to the next master file defined in the library chain. Therefore, Option 12.5 (LANG Definition Table) must specify a skeleton of NOP for the languages assigned to bind options members during initial logout. NOP indicates that no compile JCL skeleton is executed at login (the member is simply moved into the next master file). However, if you want additional processing to take place during login of a bind options member, you can specify an actual JCL skeleton in Option 12.5.

  Note: CCF Option 0 (CCF PARMS) allows for the specification of only one DB2 for z/OS and OS/390 subsystem. It is possible to accomodate multiple DB2 for z/OS and OS/390 subsystems at a site. This can be done by creating a separate skeleton for each language and DB2 for z/OS and OS/390 subsystem, and hard coding the subsystem in each flavor of the skeleton.
- The Option 12.0 (PROFILE) keyword PMOVEMENT determines who (programmer or control group) moves the DB2 for z/OS and OS/390 source program into production. Regardless of who performs the actual login, LIB/CCF submits a batch job to perform the precompile and compile.

For ISPF

Skeleton $CCFJ030 is used for Assembler. $CCFJ031 is used for COBOL. $CCFJ032 for PLI.

For CA Roscoe

Skeletons $CHGJ030, $CHGJ031 and $CHGJ032 are used.

However, you can supply a JCL skeleton that performs any combination of the precompile, compile, bind, and link. Therefore, Option 12.5 (LANG Definition Table) should specify as many language/JCL skeleton combinations as needed to perform the DB2 for z/OS and OS/390 processing.

- From Option 6 (LINK), the programmer can choose between either linking or binding an application. When he chooses the bind option, the System Bind panel displays, prompting him for either a BIND SYSIN member or a complete JCL member. When he specifies the BIND SYSIN member, LIB/CCF submits the default bind skeleton ($CCFJ103 for ISPF, $CHGJ103 for CA Roscoe) using the specified member as the bind input. Optionally, you can specify an override skeleton from this panel to override the default bind skeleton. When you specify a JCL member, that member is submitted. When you perform the bind online, you can only specify the BIND SYSIN member. For details on bind processing, see Initiating a Production Bind Request in the *LIB/CCF User Guide.*

  Note: You can control use of the override skeleton through Exit 13. See the section in this guide on LIB/CCF exits for your particular system.

  When a programmer issues a bind request from Option 6, the request is placed in the Option 10 (LINK STATUS) queue. A bind request can be differentiated from a link request by examining the request record for bind-specific information, such as the BIND SYSIN member or BIND JCL member to use.

- In Option 12.11.1 (Create Library Chain), the Associated Files panel includes the field DBRM LIBRARY. Fill in this field with the PDS data set name that contains the output from the precompile (DBRMs). This data set name is substituted for the &IMTD variable in any precompile or BIND JCL skeleton.

- You must specify the DB2 for z/OS and OS/390 subsystem name in Option 0 (CCF PARMS) in the job card specification panel (in the field DB2 SUBSYSTEM). The default name is DSN.

- Online DB2 for z/OS and OS/390 reports are available for CA Roscoe and TSO from Option 8.10. Use of the reports is optional. However, if used, you must first create a DB2 for z/OS and OS/390 table and bind the plans that update the table and generate the reports. See the DB2 for z/OS and OS/390 LIB/CCF Audit Trail section in this chapter for details on creating the table and plans. See the *LIB/CCF User Guide* for sample DB2 for z/OS and OS/390 reports.

- If your site does not want to use the Option 8.10 DB2 reports, you must modify the distributed precompile JCL skeletons by removing the step that updates the DB2 for z/OS and OS/390 table the reports use.

- Specify DB2PKG YES in Option 12.0 (PROFILE) to enable support for DB2 release 2.3.

DB2 for z/OS and OS/390 LIB/CCF Audit Trail

To establish a link between a source program stored on a master file and a DB2 for z/OS and OS/390 plan or package, LIB/CCF maintains a DB2 for z/OS and OS/390 table containing information on each DB2 for z/OS and OS/390 source member that was precompiled/compiled through LIB/CCF. Once this table is established, you can use Option 8.10 (DB2 REPORTS) to generate various reports by obtaining information from the LIB/CCF DB2 for z/OS and OS/390 table and the IBM DB2 for z/OS and OS/390 tables, SYSIBM.SYSDBRM, SYSIBM.SYSPLAN, SYSIBM.SYSPKLIST, and SYSIBM.SYSPACKAGE (the latter two are for DB2 release 2.3).

Note the following:

- The DB2 for z/OS and OS/390 reports are available only in the z/OS and OS/390 environment.
- This facility is optional. It is not required to manage the update cycle of DB2 for z/OS and OS/390 source programs. However, use it if you want a positive link between source and plan or package.
- USERMODS must be applied during the LIB/CCF installation to request reports from Option 8.10. See the *Getting Started* for details. Establishing the link between source and plan or package requires the following:
- Create the DB2 for z/OS and OS/390 table using the supplied SQL statements.
- Bind the plans that update the DB2 for z/OS and OS/390 table and generate the reports.
- Update the table each time a DB2 for z/OS and OS/390 source member is compiled. This is accomplished with an additional step in the precompile/compile JCL skeletons.
- Use the Option 8.10 reports.

The following sections describe each of these topics.

Creating the B2 for z/OS and OS/390 Table

The CA Librarian CALJJCL library contains a member called LJDB2INT. This member contains JCL and a series of SQL statements that create the table. The following figure shows the contents of member LJDB2INT. See your DB2 for z/OS and OS/390 system administrator for assistance in creating this table.

```
//LJDBINT JOB ...........
 //*****************************************************
//* MEMBER = LJDB2INT FROM CALJJCL
//*****************************************************
//* DESC: CREATE DB2 TABLES FOR CCF.
 //*****************************************************
//STEP1 EXEC PGM=IKJEFT01,DYNAMNBR=20,COND=(4,LT)
 //SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//*-----------------------------------------------------
//* CORRECT DB2 SYSTEM ID MUST BE SUPPLIED FOR YOUR SITE
//*-----------------------------------------------------
//SYSTSIN  DD *
  DSN SYSTEM(DSN)            A
  RUN PROGRAM(DSNTIAD) PLAN(DSNTIA61)
 //SYSIN    DD *
CREATE STOGROUP LIBCCFST }
   VOLUMES (xxxxxx)      }   B
   VCAT xxxx;            }
CREATE DATABASE LIBCCFDB }
  STOGROUP LIBCCFST      }   C
  BUFFERPOOL BP0;        }
CREATE TABLESPACE LIBCCFTB  }
  IN LIBCCFDB              }
  USING STOGROUP LIBCCFST   }
    PRIQTY 30               }
    SECQTY 30               }   D
    ERASE NO                }
  LOCKSIZE ANY              }
  BUFFERPOOL BP0            }
  CLOSE NO;                 }
CREATE TABLE CALIBCCF.LIBCCFTA               }
      (DBRMNAME CHAR(8)             NOT NULL, }
       DBRMTIME CHAR(8)             NOT NULL, }
       DBRMLIB  CHAR(44)            NOT NULL, }
       SRCNAME  CHAR(8)             NOT NULL, }
       SRCLIB   CHAR(44)            NOT NULL, }   E
       SRCTIME  TIMESTAMP           NOT NULL, }
       WO       CHAR(8)             NOT NULL, }
       PROGR    CHAR(8)             NOT NULL, }
       APP      CHAR(20)            NOT NULL, }
       SYSTEM   CHAR(20)            NOT NULL) }
  IN LIBCCFDB.LIBCCFTB;                       }
CREATE UNIQUE INDEX CALIBCCF.LIBCCFIA }
  ON CALIBCCF.LIBCCFTA                 }
       (DBRMNAME ASC,                  }
        DBRMTIME ASC,                  }
        DBRMLIB ASC)                   }   F
```

```
  USING STOGROUP LIBCCFST            }
   PRIQTY 4                          }
   ERASE NO                          }
 BUFFERPOOL BP0                      }
 CLOSE NO; }
/*
```

- A—Specify the correct DB2 for z/OS and OS/390 subsystem name using the SYSTEM keyword.
- B—Create a storage group called LIBCCFST. You can use any name for the storage group. Alternatively, you can use an existing storage group. Supply a valid volume and VCAT.
- C—Create a database called LIBCCFDB. You can use any name for the database. Alternatively, you can use an existing database.
- D—Create a table space called LIBCCFTB. You can use any name for the table space. Alternatively, you can use an existing table space.
- E—Create a table called CALIBCCF.LIBCCFTA. You must use this name. Do not alter the SQL statements that create the table. The next section documents table fields.
- F—Create a table index called CALIBCCF.LIBCCFIA. You can use any name for the index. Use of the index is not required, but is recommended for efficiency.

Binding the Plans

The CA Librarian CALJJCL library contains a member, LJCCFBND for CA Roscoe and LUCCFBND for (ISPF)TSO, that binds three plans and GRANT EXECUTE authority on the plans to PUBLIC. Before submitting either JCL member, you must modify the JCL to conform to your site's requirements. A default DB2 for z/OS and OS/390 subsystem name of DSN is used in this JCL. PUBLIC EXECUTE authority is not required. The alternative is to supply individual EXECUTE authority for any of the LIB/CCF users that require it. Consult the *IBM Database 2* SQL *Reference Guide* for details on the GRANT command. The following table lists the names of the plans used:

DB2 for z/OS and OS/390 Plan Name	LIB/CCF System	Function
CHGPL101	ROSCOE	Generate Reports
CCFPL101	TSO	Generate Reports
CCFPL105	ROSCOE and TSO	Update DB2 for z/OS and OS/390 Table
CCFPL106	ROSCOE and TSO	Maintain DB2 for z/OS and OS/390 Table
CHGPL112	ROSCOE	Generate Reports (2.3)

DB2 for z/OS and OS/390 Plan Name	LIB/CCF System	Function
CCFPL112	TSO	Generate Reports (2.3)

Note: When applying any maintenance that requires rebinding any of these DB2 for z/OS and OS/390 plans (for example, any PTF packaged with an object module and DBRM), specify an ACTION of REPLACE.

Updating the CALIBCCF.LIBCCFTA Table

Once the CALIBCCF.LIBCCFTA table is created, it is ready for updates. LIB/CCF is distributed with three precompile/compile JCL skeletons:

- For ISPF, $CCFJ030 for assembler, $CCFJ031 for COBOL, and $CCFJ032 for PLI.
- For CA Roscoe, $CHGJ030, $CHGJ031, and $CHGJ032.

Each skeleton contains a step after the precompile that conditionally executes plan CCFPL105. Because of skeleton logic, this step gets executed only when the member is precompiled from the production master file. It does not execute when the member is precompiled from a QA master file. CCFPL105 reads the DBRM that the precompile created to extract DB2 for z/OS and OS/390-specific information. It then obtains CA Librarian-specific information from substituted variables in the JCL skeleton. Finally, it uses the File Access Interface Routines (FAIR) to obtain further information from the production master file. After gathering this information, it creates an entry for the DBRM in the CALIBCCF.LIBCCFTA table.

The table fields are:

DBRMNAME

Specifies the one- to eight-character DBRM name.

DBRMTIME

Specifies the time stamp in DB2 for z/OS and OS/390 internal format.

DBRMLIB

Specifies the data set name of the library where the DBRM resides.

SRCNAME

Specifies the one- to eight-character CA Librarian member name containing the source that created the DBRM.

SRCLIB

Specifies the production master file where the CA Librarian member (SRCNAME) resides.

SRCTIME

Specifies the date and time of the last update to the production CA Librarian member in the format YYYY-MM-DD-hh.mm.ss.mmmmmm. Any field in the time format that is not applicable to the CA Librarian member is filled in with zeros. For example, if a CA Librarian member was never updated, LIB/CCF uses the date that the member was added to the master file and the time fields contain zeros.

WO

Specifies the number of the change request.

PROGR

Specifies the one- to eight-character programmer name (the user ID or the CCFID if Option 12.8 (CONVERSION) is converting the user ID) to whom the change request is assigned.

SYS

Specifies the LANGUAGE (per CCF Option 12.5, Language Definition Table) associated with the member. This allows for the use of multiple DB2 for z/OS and OS/390 subsystems.

APP

Specifies the LIB/CCF application under which the change request was opened.

Maintaining the CALIBCCF.LIBCCFTA Table

The CALIBCCF.LIBCCFTA table can eventually contain obsolete table rows (entries). For example, table rows include the date and time of the last update (SRCTIME) to the source member (SCRNAME). If the source is updated, precompiled, and compiled again through LIB/CCF, the previous row for that DBRM is obsolete. Similarly, if a plan is deleted from DB2 for z/OS and OS/390 or a DBRM is deleted from a DBRM library, the table rows become obsolete. There is a batch maintenance utility (program $CCFB106, plan CCFPL106) to remove these obsolete table rows. Use of the maintenance utility is optional. It is recommended that you use it to keep the DB2 for z/OS and OS/390 reports concise and to maintain the size of the table. You can execute the utility at any time.

It performs the following processing:

- Reads each row in the table.
- Totals the number of DBRMs currently used in a plan. A DBRM used in more than one plan is counted only once.
- Of the DBRMs not found in a plan, totals the number of DBRMs in a DBRM library.
- Deletes each row in the table that references a DBRM that was not found in a plan or DBRM library.
- Totals the deleted rows.
- Produces a report of the totals.

You can use the following JCL to execute the maintenance utility.

```
//DB2MAINT JOB ACCT,PGMR,CLASS=0,
 //            MSGCLASS=T
//STEP1     EXEC PGM=IKJEFT01,DYNAMNBR=20,COND=(4,LT)
 //STEPLIB   DD DSN=SYS1.DSN210.DSNLOAD,DISP=SHR          A
//          DD DSN=LIBR.CCF.LOAD,DISP=SHR
//SYSTSPRT  DD SYSOUT=*
//SYSPRINT  DD SYSOUT=*
//CCFPRINT  DD SYSOUT=*                                  B
//SYSTSIN   DD *
 DSN SYSTEM(dsn)                                        C
 RUN PROG ($CCFB106) -
   PLAN (CCFPL106)
```

- A—If not in a LNKLST library, a STEPLIB is required for the DB2 for z/OS and OS/390 and LIB/CCF programs.
- B—The report output is written to the CCFPRINT DD statement.
- C—Specify your DB2 for z/OS and OS/390 subsystem name in the parentheses on the DSN SYSTEM statement.

Chapter 10: LIB/CCF-CA Earl Interface

In addition to the reporting facilities provided by Option 8 (REPORTS) and the batch change request status program ($CCFB101 for z/OS and OS/390, CCFB101 for VSE/ESA), you can generate customized reports using Advantage CA-Earl. The LIB/CCF-Advantage CA-Earl interface routine provides access to tracking information and documentation LIB/CCF collects and stores in the System Master file. You can tailor report formats and content to suit site requirements. The information available to an Advantage CA-Earl report program is mapped out in member LJEARLFD. Computer Associates also distributes three sample programs: LJEARLCR, LJEARLCA, and LJEARLSY. All three sample programs copy in member LJEARLFD for the file definitions.

For z/OS and OS/390, these members are distributed on the CALJJCL library. For VSE/ESA, they are distributed on the installation library. For VM/ESA, they are unloaded to the LIB/CMS system minidisk. The following sections describe the sample programs and how to execute them.

This section contains the following topics:

Sample Reports (see page 183)
z/OS and OS/390 Report Generation (see page 186)
VSE/ESA Report Generation (see page 188)
VM/ESA Report Generation (see page 191)

Sample Reports

The following are sample reports.

LJEARLCR and LJEARLCA

You can use these programs to print comprehensive summary reports on all change requests, all open change requests, all unassigned change requests, or all closed change requests. They can list the change requests for a single programmer, manager, or application system. They optionally print the analysis section for closed change requests and the status of logouts, links, moves, and batch jobs associated with each selected change request.

LJEARLCR produces a report format similar to the batch change request status report. LJEARLCA provides the same information in a different format. LJEARLCR and LJEARLCA use execution PARMs to determine the type of report generated.

The parameters are listed below:

[OPEN|UNASSIGNED|CLOSED|ALL|CRID=]

Specifies whether a single request, all requests, or only the open, unassigned, or closed requests are reported on.

OPEN

(Default.) Selects all open change requests.

UNASSIGNED

Selects all unassigned change requests.

CLOSED

Selects all closed change requests.

ALL

elects all open and closed change requests.

CRID=[WO*nnnnnn*|ZR*nnnnnn*|*n*]

Selects a single change request.

WO*nnnnnn*

An opened or assigned change request

ZR*nnnnnn*

A closed change request

n

Any type of change request.

PGMR=<pgmrname>

Prints only the change requests assigned to this programmer.

MGR=<managername>

Prints only the change requests for this manager.

APPL=<application name>

Prints only the change requests for this application system.

ANALYSIS

Prints the analysis section for closed change requests. You can only specify this option with ALL or CLOSED.

STATUS

Prints the status of logouts, links, moves, and batch jobs associated with all selected change requests.

ACTIVITY

Prints the activity records for change requests.

LJEARLSY

This sample program produces a report of the System Master file tables. Only one table can be listed per execution. LJEARLSY has no execution PARMs. Instead, the sample program contains Advantage CA-Earl program statements to indicate the LIB/CCF environment and the System Master file table. As distributed, the program defaults to CA Roscoe and the System Profile table.

To specify the environment, find the program statement:

```
SET CCFRT1 = X'00'
```

As indicated by the program comments:

- X'00'—Specifies CA Roscoe
- X'01'—Specifies ISPF(TSO)
- X'02'—Specifies AllFusion CA-Vollie
- X'03'—Specifies ISPF(VM/ESA)

Change the indicator to reflect the LIB/CCF system that you use.

To specify a System Master file table, find the program statement:

```
SET CCFRT3 = X'00'
```

As indicated by the program comments, X'00' through X'08' represent the various tables. X'04', X'06', and X'07' are applicable only to an CA Vollie System Master file.

Change the indicator to reflect the table.

z/OS and OS/390 Report Generation

The following figure contains sample JCL to execute the sample programs under z/OS and OS/390.

```
//CCFEARL  JOB programmer,class=x,
 //             MSGCLASS=x
//EARL     EXEC PGM=EARL,PARM='CRID=2,ALL,ACTIVITY,STATUS'            A
//STEPLIB  DD DSN=CAI.CALJLINK,DISP=SHR                               B
//EARLOBJ  DD UNIT=DISK,SPACE=(CYL,4,,CONTIG),                        C
//         DSN=&&OBJ,DISP=(,DELETE)
 //SYSMAST  DD DSN=LIBR.USER01.V44.SYSMAST,DISP=SHR                   D
//K
//EARLLIB  DD DSN=CAI.CALJJCL,DISP=SHR             E
//*
//SORTLIB  DD DSN=SYS1.SORTLIB,DISP=SHR }
//SORTIN   DD UNIT=DISK,SPACE=(CYL,4,,CONTIG) }
//SORTOUT  DD UNIT=DISK,SPACE=(CYL,4,,CONTIG) }                       F
//SORTWK01 DD UNIT=DISK,SPACE=(CYL,4,,CONTIG) }
//SORTWK02 DD UNIT=DISK,SPACE=(CYL,4,,CONTIG) }
//SORTWK03 DD UNIT=DISK,SPACE=(CYL,4,,CONTIG) }
//*
//WORK1    DD UNIT=SYSDA,SPACE=(CYL,1)}
 //SYSUT1   DD UNIT=SYSDA,SPACE=(CYL,1)}
 //SYSUT2   DD UNIT=SYSDA,SPACE=(CYL,1)}                              G
//SYSUT3   DD UNIT=SYSDA,SPACE=(CYL,1)}
 //SYSUT4   DD UNIT=SYSDA,SPACE=(CYL,1)}
 //SYSTU5   DD UNIT=SYSDA,SPACE=(CYL,1)}
 //SYSUDUMP DD SYSOUT=*
//SYSEARL  DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//SYSIN    DD DSN=PTNSUP1.LVL1.LIBR43.CAI.CALJJCL(LJEARLCA),DISP=SHR   H
```

- A—Specify PARM= only when using LJEARLCR or LJEARLCA as SYSIN.
- B—The STEPLIB DD statement should refer to your LIB/CCF load library and the CA Earl load library.
- C—This JCL creates a temporary object library each time it is executed. You can decrease execution time by retaining the object library and using it for subsequent executions. See the *Advantage CA-Earl Reference Guide* for details on the JCL required for this type of execution.
- D—The SYSMAST DD statement should refer to the LIB/CCF System Master file. If the SYSMAST DD statement is omitted, the System Master file is dynamically allocated based on the SYSMAST specified in the $CCFGEN macro.
- E—The EARLLIB DD statement should refer to the CA Librarian CALJJCL library containing the member LJEARLFD.
- F—Advantage CA-Earl sort files.

- G—Advantage CA-Earl work files.
- H—The SYSIN DD statement should be followed by one of the sample programs (LJEARLCR, LJEARLCA, or LJEARLSY).

VSE/ESA Report Generation

The following figure contains sample JCL to execute the sample programs under VSE/ESA.

```
// JOB jobname
// DLBL IJSYS01,'work.file.file-id1',0,SD  }
// EXTENT SYS001,volser,1,0,reltrk,ntrks   }
// ASSGN SYS001,DISK,VOL=volser,SHR        }
// DLBL IJSYS02,'work.file.file-id2',0,SD  }
// EXTENT SYS002,volser,1,0,reltrk,ntrks   }
// ASSGN SYS002,DISK,VOL=volser,SHR        }
// DLBL IJSYS03,'work.file.file-id3',0,SD  }
// EXTENT SYS003,volser,1,0,reltrk,ntrks   }
// ASSGN SYS003,DISK,VOL=volser,SHR        }
// DLBL IJSYS04,'work.file.file-id4',0,SD  }            A
// EXTENT SYS004,VOLSER,1,0,reltrk,ntrks   }
// ASSGN SYS004,DISK,VOL=volser,SHR        }
// DLBL IJSYS05,'work.file.file-id5',0,SD  }
// EXTENT SYS005,volser,1,0,reltrk,ntrks   }
// ASSGN SYS005,DISK,VOL=volser,SHR        }
// DLBL IJSYS06,'work.file.file-id6',0,SD  }
// EXTENT SYS006,volser,1,0,reltrk,ntrks   }
// ASSGN SYS006,DISK,VOL=volser,SHR        }
// DLBL SYSMAST,'lib.master.file-id',,DA   }
// EXTENT SYSnnn,volser,1,0,reltrk,ntrks   }            B
// ASSGN SYSnnn,DISK,VOL=volser,SHR        }
*
// DLBL SORTWK1,'work.file.sort1',0,SD       }
// EXTENT SYSnnn,volser,1,0,reltrk,ntrks     }
// ASSGN SYSnnn,DISK,VOL=volser,SHR          }
// DLBL SORTWK2,'work.file.sort2',0,SD       }
// EXTENT SYSnnn,volser,1,0,reltrk,ntrks     }
// ASSGN SYSnnn,DISK,VOL=volser,SHR          }          C
// DLBL SORTOUT,'work.file.earl.output',0,SD }
// EXTENT SYSnnn,volser,1,0,reltrk,ntrks     }
// ASSGN SYSnnn,DISK,VOL=volser,SHR          }
// DLBL SORTIN1,'work.file.earl.output',0,SD }
// EXTENT SYSnnn,volser,1,0,reltrk,ntrks     }
// ASSGN SYSnnn,DISK,VOL=volser,SHR          }
*
// DLBL WORK1,'work.file.earl.input',0,SD  }
// EXTENT SYSnnn,volser,1,0,reltrk,ntrks   }            D
// ASSGN SYSnnn,DISK,VOL=volser,SHR        }
*
// DLBL EARLOBJ,'earl.object.file',0       }
// EXTENT SYSnnn,volser,1,0,reltrk,ntrks   }            E
// ASSGN SYSnnn,DISK,VOL=volser,SHR        }
*
// LIBDEF SOURCE,SEARCH=(libr.install),TEMP          }
// LIBDEF PHASE,SEARCH=(prod.libccf,prod.earl),TEMP  }   F
// EXEC EARL,SIZE=256K
```

- A—Advantage CA-Earl work files.

- B—The SYSMAST DLBL statement should refer to the LIB/CCF System Master file. If the SYSMAST DLBL statement is omitted, the System Master file is dynamically allocated based on the SYSMAST specified in the $CCFGEN macro.
- C—Advantage CA-Earl sort files.
- D—If Advantage CA-Earl was installed with the SORT=SRAM option, you can use the WORK1 DLBL in place of the SORTIN1 DLBL. If the SORT=SRAM option was not specified, remove the WORK1 DLBL.
- E—Advantage CA-Earl object library. This JCL creates a temporary object library each time it is executed. You can decrease execution time by retaining the object library and using it for subsequent executions. See the *Advantage CA-Earl VSE/ESA Reference Guide* for details on the JCL required for this type of execution.
- F—The source LIBDEF statement should refer to the CA Librarian installation library containing the member LJEARLFD. The phase LIBDEF statement should refer to your LIB/CCF and Advantage CA-Earl core image library. This example assumes that these libraries are defined in standard labels.

Follow this JCL with one of the sample programs (LJEARLCR, LJEARLCA, or LJEARLSY).

VM/ESA Report Generation

The following libraries and EXECs must be available to generate the Advantage CA-Earl reports in VM/ESA.

The following libraries are created on the LIB/CCF installer's A-disk during the LIB/CCF install:

CCFEARL MACLIB

The source library containing LJEARLFD.

CCFEARL TXTLIB

The text library containing the LIB/CCF-Advantage CA-Earl interface module.

The following VM/ESA files are unloaded from the LIB/CMS installation tape to the LIB/CMS installer's A-disk:

CCFEXIT EXEC

Defines the CCFEARL TXTLIB for execution.

LJEARLCR EARL

The change request report.

LJEARLCA EARL

The alternate format change request report.

LJEARLSY EARL

The System Master file table report.

The following EXEC is provided with Advantage CA-Earl:

CAEARL EXEC

Invokes Advantage CA-Earl.

Additionally, LIB/CCF and Advantage CA-Earl must be installed.

To execute the sample programs, issue the following command:

```
CAEARL earl-pgm [PARM='parameter'] (MACLIB CCFEARL EXIT CCFEXIT
```

PARM= is valid only with LJEARLCR and LJEARLCA. You must separate parameters with a comma and enclose them in apostrophes. All parameters are optional.

Chapter 11: LIB/CCF-CA Netman Interface

A LIB/CCF-Unicenter CA-Netman interface is available to z/OS and OS/390 sites. CA Netman is an information management system that maintains an online inventory of every piece of hardware and software in an organization, including configuration, financial, problem, and change information. The LIB/CCF-Unicenter CA-Netman interface provides a communication vehicle between LIB/CCF and CA Netman so that the approval of a Unicenter CA-Netman change request triggers the initiation of a LIB/CCF change request. Conversely, the closing of a LIB/CCF change request notifies Unicenter CA-Netman of the event and provides closure information.

The interface consists of four components. They are:

$CCFB103

The Change Request Initiation program. When a CA Netman change is approved, CA Netman executes NTMLBCCF. NTMLBCCF executes $CCFB103 to open a LIB/CCF change request. The eight-character CA Netman change ID is appended to the CA Netman task ID to form the ID used for the $CCFB103 -NETM control statement. This ID is stored in the LIB/CCF change request in the OPENED: field. $CCFB103 is created during LIB/CCF installation and resides in the CA Librarian target load library.

$CCFB107

The program that writes change request closure information to CA Netman. $CCFB107 is created during LIB/CCF installation and resides in the CA Librarian target load library.

LJEARLNM (CCFN0)

The source for the CA Earl program that formats the change request closure information passed to $CCFB107. Member LJEARLNM on the CA Librarian CALJJCL library must be saved and link edited to create load module CCFN0.

EXIT14

The LIB/CCF exit invoked when a change request is closed through Option 7 (CLOSE) ($CHGU014 for CA Roscoe or $CCFU014 for TSO). If the exit finds the CA Netman ID in the change request's OPENED: field, it executes CCFN0 and $CCFB107 to pass the closure information to CA Netman.

This section contains the following topics:

Installation (see page 194)
Operations (see page 196)

Installation

Release 4.8 or higher of CA Netman is required. The interface installation consists of the following steps:

1. Define NETMAN as a LIB/CCF user authorized to open change requests. Supply a -USER *xxxxxxx* in the CA Netman $CCFB103 skeleton JCL, where *xxxxxxx* is the one- to eight-character CA Netman identifier of your choice. You must also add this identifier to LIB/CCF Option 12.1 (USERS). See the *CA Netman Techniques Guide* for details on the $CCFB103 skeleton JCL. See the User Definition Table in Chapter 3, “Administrator Functions in LIB/CCF,” for details on Option 12.1.

2. Save and link edit LJEARLNM to create CCFN0. You can use the following sample JCL.

```
//jobname JOB acct,pgmr,class=a
//EARL     EXEC PGM=EARL,REGION=6000K
A   column=46./STEPLIB DD DSN=cai.ca90s.loadlib,DISP=SHR
B   column=50./EARLLIB DD DSN=libr.caljjcl.library,DISP=SHR
C   column=51./EARLOBJ DD UNIT=DISK,SPACE=(CYL,4,,CONTIG),
 //            DSN=&&OBJ,DISP=(,PASS)
 //SYSUT1   DD UNIT=SYSDA,SPACE=(CYL,1)}
 //SYSUT2   DD UNIT=SYSDA,SPACE=(CYL,1)}
 //SYSUT3   DD UNIT=SYSDA,SPACE=(CYL,1)}
D   column=41./SYSUT4 DD UNIT=SYSDA,SPACE=(CYL,1)}
 //SYSUT5   DD UNIT=SYSDA,SPACE=(CYL,1)}
 //SYSUT6   DD UNIT=SYSDA,SPACE=(CYL,1)}
 //SYSPRINT DD SYSOUT=T
//SYSOUT   DD SYSOUT=T
//SYSIN    DD *
E   column=20.OPTION SAVE=CCFN0
COPY LJEARLNM
//LKED EXEC LKED,PARM='LIST,MAP,LET,NORENT,XREF'
//LKED.SYSLIN DD DSN=&&OBJ,DISP=(OLD,DELETE)
F   column=57./LKED.SYSLMOD DD DSN=cai.ccf.loadlib(CCFN0),DISP=SHR }
//LKED.SYSLIB DD DSN=cai.ccf.loadlib,DISP=SHR }
```

   - A—Unicenter TNG Framework for z/OS and OS/390 target load library.
   - B—CA Librarian CALJJCL library containing LJEARLNM.
   - C—Advantage CA-Earl object library used to store the object module CCFN0.
   - D—Advantage CA-Earl work files.
   - E—Advantage CA-Earl control statements to copy LJEARLNM from the CALJJCL library and save the object as CCFN0.
   - F—CA Librarian target load library.

3. Modify the CA Netman PI data set name For CA Roscoe:

Import $CHGU014 from the LIB/CCF CA Roscoe RPF master file (distributed as CAI.CALJRPF) and save it as $CHGU014 on the CA Roscoe key where LIB/CCF is installed. $CHGU014 executes the following ALLOC statement:

```
ALLOC NTMPI DSN=NET9505.R43.NTMPI,DISP=SHR
```

Modify the data set name if necessary and update the member For LIB/CCF-ISPF(TSO):

$CCFU014 executes CLIST LUCCFULS that resides in the CA Librarian CALJJCL library. The CLIST executes the following ALLOC and CALL statements:

```
ALLOC F(NTMPI) DS('NET9505.R43.NTMPI') SHR
CALL 'CAI.CCF.LOAD(CCFN0)' 'CRID=&CR,TL=65,CLOSED,ANALYSIS'
CALL 'CAI.CCF.LOAD($CCFB107)' '&NETM.T'
```

Modify the data set names if necessary and update the member. Copy the member into the CLIST library of your choice. Update the EPL (CA Roscoe only).

Verify that the CA Roscoe Eligible Program List (EPL) contains entries for $CCFB107 and CCFN0. Member LJCCFEPL on the CA Librarian CALJJCL library contains the correct values for each program.

4. Activate Exit 14. For CA Roscoe:

   The presence of the exit member in the LIB/CCF CA Roscoe library activates the exit. For LIB/CCF-ISPF(TSO):

   Specify EXIT14 YES in Option 12.0 (PROFILE). The sample $CCFU014 load module is created during LIB/CCF installation and resides in the CA Librarian target load library. The presence of EXIT14 YES in Option 12.0 activates the exit.

Operations

The following is the sequence of events in a LIB/CCF-CA Netman interface cycle:

1. A problem report or change request is approved in CA Netman, triggering the execution of NTMLBCCF and $CCFB103 to open a LIB/CCF change request.
2. Based on the application for which the change request is opened, the manager views the change request, assigns to a programmer, and the programmer performs the work.
3. When the programmer closes the change request, Exit 14 is invoked. Exit 14 determines if CA Netman opened the change request (by verifying the presence of the CA Netman ID in the OPENED: field.

   - For CA Roscoe:

     Exit 14 writes any CA Netman messages to the user's CA Roscoe member called NETMMSGS. In the event of an error message, Exit 14 displays a panel directing the user to view this member. After the cause of the error is corrected, you can re-execute Exit 14 by issuing the following command:

     pfx.$CHGU014 crid *netman-id*

     pfx

     User's prefix (necessary only if executed from another CA Roscoe key).

     crid

     Change request ID in WO*nnnnnn* or *n* format.

     netman-id

     CA Netman change and task IDs. They can be obtained from theNETMMSGS member or by viewing the change request report online (the OPENED: field) from Option 8.8.

     For example:

     ```
     $CHGU014 WO000023 CHGJ0005TSK00001
     ```

     Alternatively, you can execute a batch job to write the close information to CA Netman. See the following figure for details.

   - For LIB/CCF-ISPF(TSO):

     Exit 14 writes any CA Netman messages to the user's terminal. In the event of an error, you must correct the error and re-execute LUCCFULS CLIST by issuing the following command from the TSO READY prompt:

     LUCCFULS *crid netman-id*

     crid

     Change request ID in WO*nnnnnn* or *n* format.

     netman-id

CA Netman change and task IDs. You can obtain them by viewing the change request report online (the OPENED: field) from Option 8.8.

For example:

```
LUCCFULS WO0000023 CHGJ0005TSK00001
```

Alternatively, you can execute a batch job to write the close information to CA Netman. See the following figure for details.

You can use the following JCL to rerun CCFN0 and $CCFB107 in the event of an error:

```
     // jobname JOB acct,pgmr,class=a
A       column=34./EARL EXEC PGM=CCFN0,
     // PARM='CRID=1,TL=65,CLOSED,ANALYSIS'
B       column=46./STEPLIB DD DSN=cai.ccf.loadlib,DISP=SHR
     // SYSPRINT DD DUMMY
C       column=46./EARLOUT DD UNIT=SYSDA,SPACE=(TRK,(1,1)),
     // DISP=(,PASS),DSN=&&TEMP
D       column=47./NETMAN EXEC PGM=$CCFB107,PARM='netman-id'
     // STEPLIB DD DSN=cai.ccf.loadlib,DISP=SHR
     // EARLOUT DD DSN=&&TEMP,DISP=(OLD,DELETE)
E       column=44./NTMPI DD DSN=netman.pi.file,DISP=SHR
```

- A—The required parms for CCFN0. The CRID=n is the change request ID in WOnnnnnn or n format. Specify the TL=65, CLOSED, and ANALYSIS parms as shown.
- B—CA Librarian target load library.
- C—Temporary data set for the Advantage CA-Earl formatted information, which gets passed to $CCFB107.
- D—The netman-id parameter is required for $CCFB107. The value of netman-id is the CA Netman change and task IDs that you can obtain from the NETMMSGS member (for CA Roscoe only) or by viewing the change request report online (the OPENED: field) from Option 8.8. The parameter length is 16 characters.
- E—CA Netman PI data set.

Chapter 12: Customizing Your System: LIB/CCF-CA Roscoe

You can tailor LIB/CCF-CA Roscoe to your site's needs in a number of ways over and above the options available to the system administrator on the LIB/CCF Table Processors. A number of exit points and sample exits are available. You need to tailor JCL skeletons to your site's requirements. You can also modify panels and reports.

This section contains the following topics:

User Exits (see page 200)
JCL Skeletons (see page 207)
Link-Edit Considerations (see page 224)
Customizing Panels and Reports (see page 226)
Execution (see page 228)
Identifying Panels (see page 228)
PF Keys and the Jump Command (see page 228)
Job Card Specification (see page 229)
System Error Processing (see page 230)

User Exits

LIB/CCF executes the user exits if they are present in the LIB/CCF administrator's CA Roscoe library. Sample exit programs are provided on the CAI.CLJ41RPF file created during the CA Librarian installation. See the *Getting Started* for details.

$CHGU000

Invoked before the creation of a Module Tracking Record (MTR) during logout. It can return up to 42 bytes of user data as the R1 variable that is placed in the MTR and any Module Movement Records (MMRs) or Job Submission Records (JSRs) that can be created during the LIB/CCF cycle.

$CHGU001

Invoked before the creation of a System Link/Bind Record (SLR). It can return up to 56 bytes of user data as the R1 variable that is placed in the SLR and any Job Submission Records (JSRs) that can be created by the link or bind request.

$CHGU002

Invoked before the creation of a Module Movement Record (MMR). It can return up to 42 bytes of user data as the R1 variable that is placed on the member movement record and job submission records that can be created by the login request. If this exit is not used and $CHGU000 supplied data during the logout request, then the existing user data is carried forward to the member movement record and job submission record. The exit is passed arguments:

```
A1 = Module name
A2 = The master file where the member currently resides.
```

$CHGU003

Invoked after the data is entered and verified on the logout panel. This exit must *not* interrupt the active logout panel. Can verify data filled in on the logout panel. It has access to the data fields on the panel as follows:

- P.DATA1—Member name
- P.DATA2—Change Request ID
- P.DATA3—Production master file indicator.

It is passed arguments:

- A1—PROD master
- A2—TEST master.

The exit can place an error message in the message field P.MSG and redisplay the panel with the message by returning the name of the panel field where the cursor is placed as the R1 variable. For example:

```
LET P.MSG = 'MODULE NAME DOES NOT CONFORM TO STANDARDS'
RETURN 'P.DATA<1>'
```

$CHGU004

Invoked before the history update or create panel displays, creating an opportunity to provide data for panels $CHGPAND and $CHGPANE. It can standardize information for member history creation. The MODIFY function named below pertains to the modification of existing history information through the Option 5 HISTORY command. To suppress the display of the history create and update panel, the exit should return an asterisk as the R1 parameter.

This exit must *not* interrupt the active panel and can modify the data fields as follows.

- CREATE or MODIFY function:
  - A1—Create (C) , modify (M), or batch logout (O)
  - A2—member name
  - A3—prod master
  - A4—CR ID
  - P.DATA<1>—system
  - P.DATA<2>—description
  - P.DATA<3>—language
  - P.DATA<4> through P.DATA<19>—abstract
- UPDATE function:
  - A1—Update (U) or batch login (I)
  - A2—member name
  - A3—prod master
  - A4—CR ID
  - P.DATA<1> through P.DATA<19>—description

$CHGU005

Invoked after the history update or create panel is filled in and verified. It can verify data supplied when history creation is complete. The MODIFY function named below pertains to the modification of existing history information through the Option 5 HISTORY command. This exit must **not** interrupt the active panel, and has access to the data fields as follows:

- CREATE or MODIFY function:
  - A1—Create (C), modify (M), or batch logout (O)
  - A2—member name
  - A3—prod master
  - A4—CR ID
  - P.DATA<1>—system
  - P.DATA<2>—description

- P.DATA<3>—language
- P.DATA<4> through P.DATA <19>—abstract

- UPDATE function:
  - A1—Update (U) or batch login (I)
  - A2—member name
  - A3—prod master
  - A4—CR ID
  - P.DATA<1> through P.DATA<19>—description

The exit can place an error message in the message field P.MSG and redisplay the panel with the message by returning the name of the panel field where the cursor is placed as the R1 variable. For example:

```
LET P.MSG = 'DESCRIPTION DOES NOT CONFORM TO STANDARDS'
RETURN 'P.DATA<DATA>'
```

$CHGU006

Invoked after the login function (Option 5) is successfully completed. It can produce special reports or issue messages. The following variables are provided to the exit:

- A1—Member name
- A2—Production Master File name
- A3—Change Request ID
- A4—Test Master File Name
- A5—Command as entered on the panel

$CHGU007

Invoked before performing the member lock function. The AWS contains the update stream to lock the member (-MCD and -SEL card). The exit can change or add to this stream. This is also the point where the member can automatically be EXPORTed from the CA Roscoe library to the test library if you are using the CA Roscoe library for development. It can also supply additional data during the update to lock a member on the test master when the programmer submits a login request to the control group.

The following variables are provided to the exit:

- A1—Test Master File name.
- A2—Member name.
- A3—Member password.
- A4—Master file MCD.
- A5—CCF ID

$CHGU008

Invoked before processing the link-edit SYSLIN data or link-edit JCL member. It can examine and modify the linkage editor control statements during system link request processing. The exit can verify or modify data in the AWS. The variable is A1, User data area from SLR.

$CHGU010

Invoked by Option 9 when the PROCESS, REJECT, MOVE, or COMPILE command is entered and the movement of the member is toward production (as indicated by a TO: PROD indicator in the first line of the Module Movement Record in Option 9).

This exit restricts these commands to a subset of the control group.

If the exit rejects the command, the exit must return an error message beginning with * as the R1 value. If the exit does not return an error message beginning with * as the R1 value, CCF processes the command after returning from the exit.

The following variables are provided to the exit:

- A1—Member name
- A2—Production Master File name
- A3—Test Master File name
- A4—Change Request ID
- A5—CCF ID
- A6—User data area
- A7—Command indicator (P, R, M, or C)

$CHGU011

Invoked after processing the logout request. It can automatically IMPORT a member into the CA Roscoe library from the test library. The variables are:

- A1—Member name
- A2—Test master file name
- A3—Change Request ID
- A4—Master file MCD

$CHGU012

Invoked for any batch job submitted by LIB/CCF (with the exception of print jobs), after the JCL is tailored (that is, the variables were substituted) into the AWS and before the job being submitted. It can modify JCL before the submit. LIB/CCF creates a temporary CA Roscoe member that contains the thirty-nine (39) extension variables (TAGX001 through TAGX039) and makes them available to the exit. These variables are documented in Appendix A of this guide. The format of the CA Roscoe member is one variable per line. The exit can modify or add to the JCL stream in the AWS.

The variables are:

- A1—Function indicator (I for log in, O for log out, R for reject, L for link edit or bind, T for test).
- A2—CA Roscoe member name containing variables.
- A3—Change Request ID.
- A4—Module name processed by JCL.
- A5—The language, supplied at History Create/Modify, that determines the skeleton used at login. Applicable only for login. If the function indicator is O, R, or L, then A5 is blank.
- A6—User data area.

$CHGU013

Invoked after the SYSTEM LINK or SYSTEM BIND panel is filled in. It can control the use of the LINK or DB2 for z/OS and OS/390 bind override skeleton from Option 6 (LINK). The exit can modify the OVERRIDE SKELETON name supplied, reject it, or supply a new one. This exit must **not** interrupt the active panel. It has access to the data fields as follows.

The variables are:

- A1—CR ID
- A2—CCF ID
- A3—LINK/BIND SYSIN MEMBER (from panel)
- A4—Application name
- A5—Function indicator (L for LINK, B for BIND)
- P.DATA<4>—OVERRIDE SKELETON name (from panel)
- P.MSG—message

The exit can place an error message in the message field P.MSG and redisplay the panel with the message by returning a non-null value as the R1 variable. For example:

```
LET P.MSG = 'OVERRIDE SKELETON NOT PERMITTED'
RETURN 1
```

$CHGU014

Invoked after a change request is closed from Option 7 (CLOSE). It can interface to Unicenter CA-Netman. For more information, see Chapter 12, "LIB/CCF-Unicenter CA-Netman Interface."

The variables are:

- A1—CR ID
- A2—Unicenter CA-Netman ID
- A3—System master file ddname
- A4—System master file MCD base code

$CHGU015

Invoked when a change request is opened using Option 1 (OPEN). It can assign site-specified change request numbers.

The variables are:

- A1—CR ID
- R1—CR number to assign

Sample User Exits

$CHGU000, $CHGU001, **and** $CHGU002

These exits are similar in that they can provide user data that is placed in the LIB/CCF system records. Computer Associates provides sample exits that display a panel asking for the date and time the request should be done. It then puts this data in the user area. $CHGU001 also asks for the member name that is linked. Sample exit 8 uses this information.

$CHGU003

A sample exit that prevents certain master files from being selected for logout. For this example, these master files are quality assurance libraries that another exit uses. You can modify this exit to check the CCFID and permit only certain programmers to use certain master files or certain member names.

$CHGU004

A sample exit fills in the CREATE panel with default information. To illustrate how panel data can be passed to the next exit, the information put into the description field is saved in the library member $CCFUWK1.

$CHGU005

A sample exit that retrieves the data the previous exit put into the description field (stored in the library member $CCFUWK1). It then compares the data to the description field currently on the panel. If the user changed the description, it is changed back. An error message issued, and a return code redisplays the panel.

$CHGU006

A sample exit that issues a message notifying the manager of the application for which the Change Request is opened that a member movement request is pending, assuming that the manager is a control group member.

$CHGU007

A sample exit that inserts a -PGMR card into the update stream, automatically putting the LIB/CCF ID on the member as the programmer name when the member is locked.

$CHGU008

A sample exit that takes the member name linked from the user data area (placed there by $CHGU001) and, if supplied, the link-edit SYSLIN data or JCL member is scanned for the symbolic @MODNAME@ and substituted. If some of your members are linked in the same way, this could set up a skeleton link member that can be used for several members by substituting the appropriate member name.

$CHGU010

A sample exit that restricts the PROCESS command to the CCF administrator for the final move to production for use with Q/A libraries. You must modify the sample exit for your site. It lets members of the control group process movement requests through the Q/A chain, but it permits only the CCF administrator to process the movement request to production.

$CHGU011

A sample exit that imports and saves the test member into the CA Roscoe library.

$CHGU014

A sample exit that interfaces with CA Netman. For more information, see Chapter 12, “LIB/CCF-CA Netman Interface.

$CHGU015

A sample exit that displays a panel requesting a user-specified change request number during Option 1 (OPEN) processing.

JCL Skeletons

The JCL skeletons distributed with LIB/CCF-CA Roscoe reside in the CA Roscoe library where LIB/CCF is installed. They are:

$CHGJ000

Skeleton job card for batch submissions.

$CHGJ001

Skeleton JCL for CA Librarian Utility Copy - Production to Test.

$CHGJ002

Skeleton JCL for CA Librarian Utility Copy - Test to Production.

$CHGJ003

Skeleton JCL for Production Application System LKE JCL skeletonD.

$CHGJ004

Skeleton JCL for LIBRCOPY - Test to Production.

$CHGJ005

Skeleton JCL for Copy of Object - Test to Production.

$CHGJ006

Skeleton JCL for LIBRCOPY - Production to Test.

$CHGJ007

Skeleton JCL for Production to Backup Load COPY.

$CHGJ008

Skeleton JCL for Batch Print Request.

$CHGJ009

Skeleton JCL for Automatic Job Checkoff.

$CHGJ010

Skeleton JCL for Production Source Assembly.

$CHGJ011

Skeleton JCL for Production Source COBOL Compile.

$CHGJ012

Skeleton JCL for Production Source PLI Compile.

$CHGJT10

Skeleton JCL for Test Source Assembly.

$CHGJT11

Skeleton JCL for Test Source COBOL Compile.

$CHGJT12

Skeleton JCL for Test Source PLI Compile.

$CHGJ030

Skeleton JCL for DB2 for z/OS and OS/390 precompile and assembly. For multiple DB2 for z/OS and OS/390 subsystems, create multiple copies of the skeleton, with the name of each subsystem hard-coded in the TAGX000 field.

$CHGJ031

Skeleton JCL for DB2 for z/OS and OS/390 precompile and COBOL compile. For multiple DB2 for z/OS and OS/390 subsystems, create multiple copies of the skeleton, with the name of each subsystem hard-coded in the TAGX000 field.

$CHGJ032

Skeleton JCL for DB2 for z/OS and OS/390 precompile and PLI compile. For multiple DB2 for z/OS and OS/390 subsystems, create multiple copies of the skeleton, with the name of each subsystem hard-coded in the TAGX000 field.

$CHGJ040

Skeleton JCL for a conditional assembly and link-edit.

$CHGJ100

Skeleton JCL for Test/Production Compare Option.

$CHGJ102

Skeleton JCL for Automatic Job Checkoff (LKED jobs).

$CHGJ103

Skeleton JCL for DB2 for z/OS and OS/390 bind.

$CHGJ200

Skeleton JCL for CA MetaCOBOL.

$CHGJ201

Skeleton JCL for CA APCDOC Automated Job Documentation.

$CHGJ500

Skeleton JCL to perform batch login.

$CHGJ501

Skeleton JCL to perform batch logout.

Review each of the JCL skeletons to determine which of them requires modification to conform to your site requirements. Your LIB/CCF site does not reference every skeleton. However, it is a good practice to check each one so that, if you later update your system tables to reference different skeletons, you do not need to go back to see if anything else must be changed.

Note: To avoid S806 abends due to load modules not found, either add JOBLIB or STEPLIB DD statements to the skeletons where appropriate.

$CHGJ000

Skeleton JOB card for batch submissions.

- JNAME—JOB name
- PNAME—Programmer name
- PREFIX—CA Roscoe prefix

$CHGJ001

Skeleton JCL for CA Librarian Utility Copy production to test. Copies all archive levels of a module.

- TAG0—Batch CA Librarian name
- TAG1—Member name
- TAG2—Production master file
- TAG3—Test master file
- TAG4—Test master MCD
- TAG5—Production master MCD
- TAG6—Test master file MCD security status

$CHGJ002

Skeleton JCL for CA Librarian Utility Copy - test to production. Copies all archive levels of a module.

- TAG0—Batch CA Librarian name
- TAG1—Member name
- TAG2—Production master file

- TAG3—Test master file
- TAG4—Production master MCD
- TAG5—Test master MCD
- TAG6—Production master file MCD security status
- TAG7—Test master file member password
- TAG8—Production object library

$CHGJ003

Skeleton JCL for CA Librarian production application system LKED.

- TAG1—Member name
- TAG2—Production master file
- TAG3—Change request number
- TAG4—Production master MCD
- TAG5—Production load library
- TAG6—Production master file member password
- TAG7—Production object library
- TAG8—Production backup load library

$CHGJ004

Skeleton JCL for LIBRCOPY - test to production. Copies only the current archive level of a module.

- TAG0—Batch CA Librarian name
- TAG1—Member name
- TAG2—Production master file
- TAG3—Test master file
- TAG4—Production master MCD
- TAG5—Test master MCD
- TAG6—Production master file MCD security status
- TAG7—Test master file member password
- TAG8—Production object library

$CHGJ005

Skeleton JCL for copy of object - test to production.

- TAG1—Member name
- TAG2—Production master file

- TAG3—Test master file
- TAG4—Production master MCD
- TAG5—Test master MCD
- TAG6—Production master file MCD security status
- TAG7—Test master file member password
- TAG8—Production object library
- TAG9—Test object library

$CHGJ006

Skeleton JCL for LIBRCOPY - production to test. Copies only the current archive level of a module.

- TAG0—Batch CA Librarian name
- TAG1—Member name
- TAG2—Production master file
- TAG3—Test master file
- TAG4—Production master MCD
- TAG5—Test master MCD
- TAG6—Production master file MCD security status
- TAG7—Test master file member password
- TAG8—Production object library

$CHGJ007

Skeleton JCL for production to backup LOAD copy.

- TAG5—Production load library
- TAG8—Backup load library

$CHGJ008

Skeleton JCL for batch print requests.

- TAG1—Print DEST
- TAG2—Number of copies
- TAG3—Sysout class

$CHGJ009

Skeleton JCL step for automatic job checkoff. This skeleton executes program $CCFB100, which dynamically allocates the CCF system master file based on the SYSMAST option of the $CCFGEN macro. The $CCFGEN macro creates a load module called $CCFCOMI. Therefore, this skeleton can require a STEPLIB DD statement for the load library that contains both $CCFB100 and $CCFCOMI. This skeleton has no symbolics.

$CHGJ010

Skeleton JCL for production source assembly.

- TAG0—Batch CA Librarian name
- TAG1—Member name
- TAG2—Production master file
- TAG4—Production master file MCD
- TAG7—Production member password
- TAG8—Production object library

$**CHGJ011**

Skeleton JCL for production source COBOL compile.

- TAG0—Batch CA Librarian name
- TAG1—Member name
- TAG2—Production master file
- TAG4—Production master file MCD
- TAG7—Production member password
- TAG8—Production object library

$**CHGJ012**

Skeleton JCL for production source PL1 compile.

- TAG0—Batch CA Librarian name
- TAG1—Member name
- TAG2—Production master file
- TAG4—Production master file MCD
- TAG7—Production member password
- TAG8—Production object library

$CHGJT10

Skeleton JCL for test source assembly. Only the extension variables (documented in Appendix A) are valid for this skeleton, with the exception of &TAG0 and &TAG1.

- TAG0—Batch CA Librarian name
- TAG1—Member name
- TAGX019—Test master file
- TAGX015—Test master file MCD
- TAGX028—Test object library

$CHGJT11

Skeleton JCL for test source COBOL compile. Only the extension variables (documented in Appendix A) are valid for this skeleton, with the exception of &TAG0 and &TAG1.

- TAG0—Batch CA Librarian name
- TAG1—Member name
- TAGX019—Test master file
- TAGX015—Test master file MCD
- TAGX028—Test object library

$CHGJT12

Skeleton JCL for test source PL1 compile. Only the extension variables (documented in Appendix A) are valid for this skeleton, with the exception of &TAG0 and &TAG1.

- TAG0—Batch CA Librarian name
- TAG1—Member name
- TAGX019—Test master file
- TAGX015—Test master file MCD
- TAGX028—Test object library

$CHGJ030

Skeleton JCL for DB2 for z/OS and OS/390 precompile and assembly. For multiple DB2 for z/OS and OS/390 subsystems, create multiple copies of the skeleton, with the name of each subsystem hard-coded in the TAGX000 field.

- TAG0—Batch CA Librarian name
- TAG1—Member name
- TAG2—Production master file
- TAG4—Production master file MCD
- TAGX000—DB2 for z/OS and OS/390 subsystem name

- TAGX002—Target library type
- TAGX016—Target master file MCD
- TAGX017—History master file MCD
- TAGX020—Target master file
- TAGX021—History master file
- TAGX034—z/OS and OS/390 target object library
- TAGX038—Target DBRM library
- TAGX101—Change request number
- TAGX102—CCF ID

$CHGJ031

Skeleton JCL for DB2 for z/OS and OS/390 precompile and COBOL compile. For multiple DB2 for z/OS and OS/390 subsystems, create multiple copies of the skeleton, with the name of each subsystem hard-coded in the TAGX000 field.

- TAG0—Batch CA Librarian name
- TAG1—Member name
- TAG2—Production master file
- TAG4—Production master file MCD
- TAGX000—DB2 for z/OS and OS/390 subsystem name
- TAGX002—Target library type
- TAGX016—Target master file MCD
- TAGX017—History master file MCD
- TAGX020—Target master file
- TAGX021—History master file
- TAGX034—z/OS and OS/390 target object library
- TAGX038—Target DBRM library
- TAGX101—Change request number
- TAGX102—CCF ID

$CHGJ032

Skeleton JCL for DB2 for z/OS and OS/390 precompile and PL1 compile. For multiple DB2 for z/OS and OS/390 subsystems, create multiple copies of the skeleton, with the name of each subsystem hard-coded in the TAGX000 field.

- TAG0—Batch CA Librarian name
- TAG1—Member name

- TAG2—Production master file
- TAG4—Production master file MCD
- TAGX000—DB2 for z/OS and OS/390 subsystem name
- TAGX002—Target library type
- TAGX016—Target master file MCD
- TAGX017—History master file MCD
- TAGX020—Target master file
- TAGX021—History master file
- TAGX034—z/OS and OS/390 target object library
- TAGX038—Target DBRM library
- TAGX101—Change request number
- TAGX102—CCF ID

$CHGJ040

Skeleton JCL to conditionally assemble and link edit at login time. If the movement is to production, the batch CA Librarian is executed with output exit LIBREXIT to insert Source to Load Audit Trail (SLAT) variables and perform the link-edit into the production load library. If the movement is not to production (for example, to a quality assurance library), LIB/AM assembles the member directly from the master file without performing the link-edit.

- TAG0—Batch CA Librarian name
- TAG1—Member name
- TAG7—Production member's password
- TAGX002—Target library type
- TAGX004—Security lock status
- TAGX016—Target master file MCD
- TAGX020—Target master file
- TAGX034—z/OS and OS/390 target object library
- TAGX036—z/OS and OS/390 target load library

$CHGJ100

Skeleton JCL for test/production compare option. This skeleton executes Comparator, which is site named at installation time. You must update this skeleton with the appropriate program name.

- TAG1—Member name
- TAG2—Production master file

- TAG3—Test master file
- TAG5—Current member version
- TAG6—Production MCD
- TAG7—Test MCD

$CHGJ102

Skeleton JCL step for automatic job checkoff for batch link-edits. This skeleton executes program $CCFB100, which dynamically allocates the CCF system master file based on the SYSMAST option of the $CCFGEN macro. The $CCFGEN macro creates a load module called $CCFCOMI. Therefore, this skeleton can require a STEPLIB DD statement for the load library that contains both $CCFB100 and $CCFCOMI. This skeleton has no symbolics.

$CHGJ103

Skeleton JCL for the production DB2 for z/OS and OS/390 bind.

- TAGX038—Target DBRM library
- TAGX000—DB2 for z/OS and OS/390 subsystem name

$CHGJ200

Skeleton JCL for Advantage CA-MetaCOBOL+ execution.

- TAG1—Member name
- TAG2—Production master file
- TAG4—Production master file MCD
- TAG7—Production member password
- TAG8—Production object library

$CHGJ201

Skeleton JCL for Unicenter CA-APCDOC execution for JCL procs.

- TAG0—Batch CA Librarian name
- TAG1—Member name
- TAG2—Production master file
- TAG4—Production master file MCD
- TAG7—Production member password
- TAG8—Production object library

$CHGJ500

Skeleton JCL to perform a batch login. This skeleton has no variables. Modify the data set names as necessary.

$CHGJ501

Skeleton JCL to perform a batch logout. This skeleton has no variables. Modify the data set names as necessary.

Manipulating JCL Processing

This section describes how you can alter LIB/CCF JCL processing and sequences of events.

As supplied on the installation tape, the JCL skeletons perform certain functions in a certain order. The JCL skeletons are stored as CA Roscoe library members on the key of the LIB/CCF administrator. LIB/CCF builds jobstreams from several different skeletons by first deleting the contents of the AWS, then copying in each appropriate skeleton. By definition, these skeletons have symbolics that you must replace with actual values before you can use them as JCL. Once all the needed skeletons are copied into the AWS, LIB/CCF performs an EDIT of the symbolics, changing them to their assigned values.

For example, consider how LIB/CCF prepares the JCL for a login through LIBRCOPY. First, it fetches skeleton $CHGJ000 into the AWS, which contains the job card. Then it copies skeleton $CHGJ004, which has three parts.

- The first part is labeled step GETTEST, the first half of LIBRCOPY.
- The second part is labeled step PUTPROD, the second half of LIBRCOPY
- The third part is labeled step DELTEST, to delete the member from the test master file.

LIB/CCF next copies the appropriate JCL compile skeleton, which it determines by obtaining the language type for the member from the History Master file, then using this to search the LIB/CCF Language Definition Table. Let us look at skeleton $CHGJ010, which does assemblies. There are two parts.

- The first selects the member from the production master file and writes it to OSJOB.
- The second part assembles the member.
- for more information

Finally, LIB/CCF appends the JSR Checkoff step to the very end, skeleton $CHGJ009. Next, the EDIT to translate the symbolics is performed. Then the contents of the AWS are submitted.

Once you understand the mechanics of how LIB/CCF builds JCL streams, it is clear that, by manipulating the contents of the individual skeletons, job streams can be built in any way. The following sections examine two common scenarios.

Conditional JCL Skeleton Generation

LIB/CCF CA Roscoe supports JCL skeleton processing statements that let you:

- Include JCL based on a simple condition, and
- Imbed other CA Roscoe JCL skeleton members from the LIB/CCF administrator's library.

This process occurs before any batch job submitted from LIB/CCF (with the exception of a batch print job).

The processing statements are as follows:

)SEL <*expression*> and)ENDSEL

The)SEL and)ENDSEL statements constitute an "if construct." The expression is evaluated for a true or false condition. If true, the records between the)SEL and)ENDSEL are included. If false, control is transferred to the)ENDSEL statement and these records are excluded. Unlimited nesting is supported. A)SEL statement can be up to 255 bytes long, but must be contained on one line.

The expression can consist of a comparison or combination of comparisons joined by connectors. For example:

```
V1 OPERATOR V2 CONNECTOR V3 OPERATOR V4
```

Note: If the value (V*n* in the above example) is null, blank, or contains an imbedded blank, you must enclose it in apostrophes.

The operators are as follows:

- EQ or = (Equal)
- NE or ¬= (Not equal)
- GT or > (Greater than)
- LT or < (Less than)
- LE or <= (Less than or equal)
- GE or >= (Greater than or equal)
- NG or ¬> (Not greater than)
- NL or ¬< (Not less than)

The connectors are:

- & (And)
- | (Or)

)CM

Indicates that a line is a comment and is not included in the submitted job.

)TRACE

Indicates that the)SEL and)IM constructs are traced through a panel display of the statement with the variables substituted before the statement being processed. Use the)TRACE statement when debugging.

)IM member

Indicates that the specified CA Roscoe member is imbedded from the LIB/CCF administrator's library, replacing the)IM statement. *Do not* specify the member prefix. The imbedded member can contain other processing statements. Only variables TAG0, TAG1, and TAGX000 through TAGX103 are substituted in embedded members. Computer Associates documents the extension variables (TAGX000 through TAGX103) in Appendix A of this guide.

Example

The following example demonstrates how you can use conditional logic in a SEL/ENDSEL construct to determine what type of library a module is moved into and what method to use to assemble it. The comment lines describe the conditional logic.

```
//* SKEL:  SAMPLE
//* DESC:  JCL skeleton to conditionally assemble and link
//*-------------------------------------------------------------
//* TAG0=BATCH LIBRARIAN NAME
//* TAG1=MODULE NAME
//* TAG7=PRODUCTION MODULE PASSWORD
//* TAGX002=TARGET LIBRARY TYPE
//*         (T=TEST, P=PRODUCTION, R=REJECT, Q=Q/A)
 //* TAGX004=SECURITY LOCK STATUS
//* TAGX016=TARGET MASTER MCD
//* TAGX020=TARGET MASTER FILE
//* TAGX034=TARGET object LIBRARY
//* TAGX036=TARGET PRODUCTION LOAD LIBRARY
)CM ***********************************************************
)CM If the module movement is to production, execute a batch
)CM LIBRARIAN step before the assembly in order to use
)CM LIBREXIT to insert source to load audit trail information
)CM into the production load library.
)CM ***********************************************************
)SEL TAGX002 = P
//STEP1   EXEC PGM=TAG0,PARM='NRJS,NJTA,OEX=LIBREXIT'
//SYSPRINT  DD SYSOUT=K
//MASTER    DD DISP=SHR,DSN=TAG2
//OSJOB     DD DSN=&&TEMP,DISP=(,PASS),SPACE=(CYL,(1,1),RLSE),
 //            DCB=(DSORG=PS,BLKSIZE=3200,LRECL=80,RECFM=FB)
 //SYSIN     DD *
-OPT UTILITY
)CM ***********************************************************
)CM If there is a 'to' library MCD code, supply a -MCD card
)CM ***********************************************************
)SEL 'TAGX016' ¬= ' '
-MCD TAGX016
)ENDSEL
-EXTRACT TAG1
)ENDSEL
//STEP2   EXEC ASMHC,PARM='DECK',COND=(0,NE)
 //SYSPUNCH  DD DSN=TAGX0.34(TAG1),
 //             DISP=SHR
)CM ***********************************************************
)CM If the module movement is to production, use the &&TEMP
)CM created in STEP1 as the SYSIN file for the assembly
)CM ***********************************************************
)SEL TAGX002 = P
//SYSIN     DD DSN=&&TEMP,DISP=(OLD,DELETE)
)ENDSEL
)CM ***********************************************************
)CM If the module movement in not to production, use LAM
)CM to assemble the module directly from the master file.
)CM ***********************************************************
```

```
)SEL TAGX002 ¬= P
)CM ***********************************************************
)CM If no 'TO' library MCD exists or the 'to' library lock
)CM status is not PROD2, supply SYSIN without MCD code.
)CM ***********************************************************
)SEL 'TAGX016' = ' '& 'TAGX004' ¬= 2
//SYSIN     DD DSN=TAGX020(TAG1),SUBSYS=(LAM,'PSWD=TAG7')
)ENDSEL
)CM ***********************************************************
)CM If 'TO' library MCD exists and the 'to' library lock
)CM is PROD2, supply SYSIN with the MCD code.
)CM ***********************************************************
)SEL 'TAGX016' ¬= ' ' & 'TAGX004' = 2
//SYSIN     DD DSN=TAGX020(TAG1),SUBSYS=(LAM,'PSWD=TAG7',
 //             'MCD=TAGX016')
)ENDSEL
)ENDSEL
)CM ***********************************************************
)CM If the module movement is to production, execute the link
)CM editor using the production object and load libraries. If
)CM the movement is not to production, don't perform link.
)CM ***********************************************************
)SEL 'TAGX002' = P
//STEP3  EXEC PGM=IEWL,PARM='LIST,XREF,RENT',COND=(0,NE)
 //SYSPRINT DD SYSOUT=*
//SYSUT1   DD UNIT=SYSDA,SPACE=(1024,(50,20))
 //SYSLIN   DD *
INCLUDE RELOC(TAG1)
 ENTRY TAG1
NAME TAG1(R)
 //RELOC   DD DSN=TAGX034,DISP=SHR
//SYSLIB  DD DSN=TAGX034,DISP=SHR
//SYSLMOD DD DSN=TAGX036,DISP=SHR
)ENDSEL
```

Compile from the Test Master File

During login, some sites want the member compiled before it is moved from test to production. This means compiling from the test master file. To do this, you must do some minor changes to the JCL skeleton $CHGJ004 and the compile skeleton for the appropriate language type. This example uses the assembler skeleton, $CHGJ010.

Remember that skeleton $CHGJ004 has three logical parts to it, steps GETTEST, PUTPROD, and DELTEST. Each of these steps must be executed after the compile is performed. Move these steps to the end of skeleton $CHGJ010, following STEP2, the assembly step. Leave at least one comment statement in $CHGJ004.

The first part of $CHGJ010 selects the member from the production master file. Change the symbolic TAG2 to TAG3 to have it select the member from the test master file instead.

You must make these changes for each language type that requires this processing.

Compile and Link at Login

Because many programs consist of only one member, sites often want to compile and link at login instead of in separate steps. This saves the programmer the extra step of submitting a System Link Request. To do this, you can make modifications to the compile JCL skeletons for each language type.

Using an assembler example, append an IEWL step to the assembly step in skeleton $CHGJ010. The skeleton now has three steps; extract (STEP1), assembly (STEP2), and link (STEP3). In the assembly step, change &Pobject to &IMTO (the production object library) on the SYSPUNCH DD. In the link step, specify TAGX034 for the DSN on the object DD. Then, on the SYSLIB and SYSLMOD DDs, specify TAGX036 (the production load library) for the DSN.

The z/OS and OS/390 object LIBRARY specified in LCDF (Option 12.11) for the production master file is substituted in the JCL skeleton for the TAGX034 variable. Similarly, the z/OS and OS/390 LOAD LIBRARY is substituted for the TAGX036 variable.

Alternatively, you can eliminate the use of a permanent object library since the assembly and link are performed in the same job. If so, instead of using the TAGX034 variable for the object library, specify a temporary data set for the SYSPUNCH DD in the assembly step. Refer to that data set as the object DD in the link step.

Keep in mind that, since the link-edit is included in the login step, the programmer can no longer execute the LIB/CCF System Link Request (Option 6). As a result, the programmer is not prompted for link-edit options like NCAL, LIST, and so on, at the time the link-edit is submitted. They must be hardcoded into the skeleton.

Note the following:

- For additional information on variables, see Appendix A.
- JCL skeleton $CHGJ040 conditionally performs an assembly and link-edit based on whether the module is moved into a Q/A or production library. It also uses LIB/AM and the output exit LIBREXIT.

Link-Edit Considerations

Three Option 12.0 (PROFILE) keywords affect the way link-edits are performed in LIB/CCF. They are CLINK, LKED, and LNKNTEGRTY. See the section titled System Profile Table in Chapter 3, “Administrator Functions in LIB/CCF,” for details on each keyword.

From Option 6 (LINK), the programmer is prompted for the LIB/CCF Change Request ID and the function to perform (a link or DB2 for z/OS and OS/390 bind). For details on the DB2 for z/OS and OS/390 bind process, see the section titled Initiating a Production Bind Request in the *LIB/CCF User Guide*.

The panel defaults to L for link. The programmer sees a list of production master files and their associated production load libraries, as shown below in the sample panel. The programmer must then enter the number of the pair that defines the master file containing the associated linkage editor input and the load library where the modules should be linked.

```
THU mm/dd/yy ----------------- SYSTEM LINK EDIT ---------------- 15.42.44

COMMAND ===>

PRODUCTION MASTER FILE  ===> 001                REQUEST ID ===> WO000004

      LKED SYSIN MEMBER ===>             OR LKED JCL MEMBER ===>
      OVERRIDE SKELETON ===>

AVAILABLE FILES: CHAIN DESC./CHAIN NUMBER/PRODUCTION/LOAD:

001) PAYROLL CHAIN                                    (00001)
         LIBR.PROD.MAST
             LIBR.PROD.LOAD

002) ACCOUNTING CHAIN                                 (00002)
         LIBR.PROD.MAST2
             LIBR.PROD.LOAD2
*** END OF LIST ***
```

The programmer can enter:

- Just a LKED SYSIN MEMBER that uses the default skeleton if batch link-edits are performed,
- Just a LKED JCL MEMBER that provides a complete JCL stream,
- Or both a LKED JCL MEMBER and an OVERRIDE SKELETON that are used in place of the default skeleton.

The following pages provide information about each of these fields.

LKED SYSIN MEMBER

This field:

- Is mutually exclusive with LKED JCL MEMBER.
- Must reside on the production master file selected in Option 6.
- Must contain the required link-edit INCLUDE statements that reference a ddname of object and a link-edit NAME statement specifying the REPLACE(R) option. Other link-edit control statements are optional as required.
- Is for batch link-edits, this member is used as SYSLIN to the default link-edit skeleton ($CHGJ003) or, if specified, the override link-edit skeleton.
- Is for online link-edits, this member is used as input to an execution of IEWL run under ETSO.
- Displays the Linkage Editor Options Selection panel. To choose an option, place an **X** next to it on the display. Selected options are inserted into the PARM= field of the EXEC statement in the JCL stream that is submitted.
- Automatic load module backup (if specified in LCDF) is based on load module names on the link-edit NAME statements.

LKED JCL MEMBER

This field:

- Mutually exclusive with LKED SYSIN MEMBER and OVERRIDE SKELETON.
- Must reside on production master file selected.
- Must be a complete link-edit JCL stream.
- LIB/CCF submits as is for execution.
- Job name is derived from the job card included in this member.
- Automatic load module backup is not performed unless a step is inserted into the JCL stream to accomplish it.

OVERRIDE SKELETON

This field:

- Mutually exclusive with LKED JCL MEMBER.
- Must reside in the CA Roscoe library where LIB/CCF is installed.
- Requires that you supply a LKED SYSIN MEMBER.
- Specifies a skeleton to use in place of the default link-edit skeleton ($CHGJ003).
- Automatic load module backup (if specified in LCDF) is based on load module names on the link-edit NAME statements.

Note the following:

- If online link-editing is specified as an installation option (Option 12.0 specifies LKED ONLINE), only the LKED SYSIN MEMBER field appears on this panel.
- Sample user exits allow the use of a member name variable (@MODNAME@) in place of the actual member name in the LKED SYSIN MEMBER and LKED JCL MEMBER. See the section titled Sample User Exits earlier in this chapter for details on exits $CHGU001 and $CHGU008.

Customizing Panels and Reports

LIB/CCF is distributed with an Advantage CA-Earl (Easy Access Report Language) interface. Advantage CA-Earl is a report generation system that uses a high-level programming language to produce customized reports. The LIB/CCF-Advantage CA-Earl interface provides access to information that LIB/CCF collected and stored on the System Master file. See the “LIB/CCF-Advantage CA-Earl Interface” chapter for details.

In addition to the LIB/CCF-Advantage CA-Earl interface, Computer Associates provides a macro ($CCFCOMT) to allow modification to LIB/CCF panel and report titles. After assembling and link editing the macro with the overrides, *both* the panel titles and the report headings are affected (that is, it is not necessary to modify the actual LIB/CCF panels). Reports printed online from LIB/CCF and by the batch change request status report program ($CCFB101) reflect titles that the macro changed. The following table describes the panel titles that the macro can modify (other titles on the panels are not modifiable).

LIB/CCF Option	LIB/CCF Panel	Title Field	Macro Label	Max Length
1-OPEN	$CHGPAN1	REQUESTOR	TITLE1	11
2-ASSIGN	$CHGPAN2	PHONE	TITLE2	11
3-DISPLAY	$CHGPAN3	MANAGER PHONE DEPARTMENT COORDINATOR PHONE	TITLE3 TITLE4 TITLE5 TITLE6	11 11 11 11
2-ASSIGN	$CHGPAN2	MGR COMMENT	TITLE7	11
3-DISPLAY	$CHGPAN3			
7-CLOSE	$CHGPAN7	ANALYSIS	TITLE8	8
7-CLOSE	$CHGPANB	ROUTINES AFFECTED	TITLE9 TITLE10	11 11

LIB/CCF Option	LIB/CCF Panel	Title Field	Macro Label	Max Length
7-CLOSE	$CHGPANC	SYSTEMS TESTS MODIFIED ON SYSTEMS TESTS VERIFIED ON SYSTEMS TESTS ADDED OR MOD NEW USER INPUT NEW USER OUTPUT DOCUMENTATION CHANGES	TITLE11 TITLE12 TITLE13 TITLE14 TITLE15 TITLE16	25 25 26 14 15 21
4-LOGOUT	$CHGPAND	SYSTEM	TITLE17	11
5-LOGIN	$CHGPANE	DESCRIPTION ABSTRACT	TITLE18 TITLE19	11 11
5-LOGIN	$CHGPANE	DESCRIPTION OF CHANGES	TITLE20 TITLE21	11 11

You must assemble the $CCFCOMT macro on the CA Librarian target macro library (CAI.CAIMAC) and link it to specify title overrides. Computer Associates provides sample JCL in member LJCCFCMT on the CA Librarian CALJJCL library.

Note: The title appears on the panels and in the reports *exactly* as entered in the override. It can be right or left justified, contain embedded blanks, and be lower or upper case. You must delimit titles with apostrophes.

Additionally, you can modify the following panels as indicated below:

$**CHGPAN0**

Primary LIB/CCF panel. This is the panel where you select the LIB/CCF functions. You can modify this panel to remove options that your site does not use.

HELP PANEL: $CHGH005

$**CHGPAN8**

You can modify this panel to display any combination of the reports that this panel displays. You cannot add reports to this panel.

HELP PANEL: $CHGH080

$C**HGPN00**

This is the Option 6 panel for the LKED options specification. You can modify this panel to specify different default options.

HELP PANEL: None.

$**CHGPANP**

Panel for the LIB/CCF SYSTEM ERROR NOTIFICATION. Modify this panel to reflect the telephone number of the LIB/CCF system administrator.

HELP PANEL: $CHGH008

$CHGPANQ

Panel for LIB/CCF system error notification when an error occurs while attempting to notify the system administrator of an error. Modify this panel to reflect the telephone number of the LIB/CCF system administrator.

HELP PANEL: $CHGH008

$CHGPN01

Primary LIB/CCF panel for the system administrator. This is the panel where you select the LIB/CCF functions. You can modify this panel to remove options that your site does not use. This panel differs from $CHGPAN0 in that it includes Option 12 - LIB/CCF Administrator Functions.

HELP PANEL: $CHGH005

Execution

After LIB/CCF is installed, you can execute it by typing the name of the entry program: CCF

Optionally, you can make an option specification as follows: CCF 3 or CCF 8.2

Identifying Panels

To identify a LIB/CCF panel, enter the PANELID command on the command line of the primary menu. This displays the LIB/CCF panel name in the upper, left corner of the panel, replacing the day and date that was previously displayed. To remove the panel name, enter the **PANELID OFF** command on the command line of the primary menu.

PF Keys and the Jump Command

PF key definitions for the LIB/CCF system are non-modifiable and are set as follows:

- PF1 and PF13 (- HELP)
- PF3 and PF15 (- END)
- PF5 and PF17 (- NEXT)
- PF7 and PF19 (- SCROLL BACKWARD, BACK (Tutorials))
- PF8 and PF20 (- SCROLL FORWARD)
- PF10 and PF22 (- TOP)
- PF11 and PF23 (- BOTTOM)

Computer Associates supports a jump command that enables a direct jump to a specified option.

This command has the following format:

```
=option<.option>
```

For example, you could enter =3 or =8.5.

Job Card Specification

The JCL job statement skeleton is used when LIB/CCF jobs are submitted for batch execution. You should appropriately modify the job statement using Option 0 to reflect the correct accounting information, job class, and message class. Once a job is submitted from the CCF user's key, an CA Roscoe member is created in the user's CA Roscoe library that contains the job card used for subsequent jobs. The CA Roscoe member name is $*pfx*J000, where *pfx* is the prefix of the key where CCF is installed. There are four variables available from Option 0 that are filled in when a batch job is submitted from LIB/CCF. Use of these variables is optional.

The variables are:

JNAME

LIB/CCF generates a jobname of *pfx*CCF, where *pfx* is the user's CA Roscoe prefix.

%

The job name identifier character (A through Z) that is appended to the end of the job name variable (JNAME). This character is incremented for each job that the user submits. For example, the first job a user submits with a prefix of MIN is named MINCCFA. The second job is MINCCFB, and so on.

PNAME

CA Roscoe key.

PREFIX

CA Roscoe prefix.

The job statement can consist of up to ten lines and must begin on the first line. Any trailing unused lines are ignored.

System Error Processing

For CA Roscoe CCF, the errors can be severe. Severe errors are those that only the LIB/CCF administrator can resolve. When such an error occurs, LIB/CCF sends a message to the LIB/CCF administrator as a member in the CA Roscoe library.

For all other errors, LIB/CCF sends the messages to a member in the user's CA Roscoe library.

If you are an CA Roscoe user, find out what messages were sent to you by looking in your ZZZZZMSG member.

Should a LIB/CCF-CA Roscoe system error occur, an error panel displays showing the program and function where the error occurred. The LIB/CCF system administrator is notified of the error automatically. Contact him for an explanation of the problem and resolution. You should not try to continue using LIB/CCF until the problem is resolved. Should an error occur while attempting to notify the administrator, the error panel notifies you of this. Contact the administrator at once if this happens and relay the information on the error panel.

Chapter 13: Customizing Your System: LIB/CCF-ISPF(TSO)

You can tailor LIB/CCF-ISPF(TSO) to your site's needs in a number of ways over and above the options available to the system administrator in the Option 12 tables. A number of exit points and sample exits are available. You need to tailor JCL skeletons to your site's requirements. You can also change panels and reports.

This section contains the following topics:

User Exits (see page 231)
JCL Skeletons (see page 244)
Link-Edit Considerations (see page 256)
Customizing Panels and Reports (see page 258)
Execution (see page 259)
PF Keys and the Jump Command (see page 260)
Job Card Specification (see page 260)
System Error Processing (see page 261)

User Exits

LIB/CCF-ISPF(TSO) lets you supply exit programs that are invoked at defined points to perform site-dependent processing.

LIB/CCF executes the user exits if indicated in the LIB/CCF System Profile table (Option 12.0 keyword EXIT*n*, where *n* is the number of the exit). LIB/CCF can load the exits or link them as specified in the $CCFGEN macro (EXITn=LINK/LOAD) at installation. Computer Associates provides sample exits in the target source library (with a default name of CAI.CAISRC) in source and in the CA Librarian target load library as load modules.

The exits must be re-enterable and re-useable.

If the exit acquires storage, then it must release it before returning to LIB/CCF. However, if your exit needs to access and update data that must be preserved across exit calls, it should take advantage of the work area supplied for exits 7, 8, 9, and 10. LIB/CCF supplies a clean work area the first time it calls any one of the four exits. For subsequent calls, LIB/CCF passes the address of the work area. LIB/CCF itself does not modify the work area. Therefore, your exit can use the work area in any way you want it to.

The exit program can SAVE and RESTORE the ISPF display environment when the exit displays a panel. Failure to do so can cause dialog errors during the display of CCF panels. The following ISPF dialog services preserve the original display environment:

- CONTROL,DISPLAY,SAVE
- CONTROL,DISPLAY,RESTORE

See sample exits $CCFU000, $CCFU001, or $CCFU002 for an example of this processing.

Code all LIB/CCF exit routines using standard OS linkage conventions.

On entry, the contents of the registers are as follows:

- Register 13—Address of an 18 fullword save area
- Register 14—Returnaddress
- Register 15—Entryaddress
- Register 1—Address of a parameter list

On return:

- Register 15—Returncode

The following section summarizes the details of the parameter lists passed to each exit routine and the meaning of the return codes each exit uses.

$CCFU000

Invoked before the creation of a Module Tracking Record (MTR) when a member is logged out. The exit can supply up to 42 bytes of data to place on the MTR. Unless changed later, LIB/CCF duplicates this data on any subsequent MMRs and login JSRs for this member.

The PARM List Offset is as follows: +0—The address of the User Data Area

$CCFU001

Invoked before the creation of a System Link Record (SLR). When the programmer initiates a production link request, LIB/CCF calls exit 1, which can supply up to 56 bytes of data to place on the SLR that LIB/CCF creates. LIB/CCF duplicates this information on any subsequent JSRs if the link-edit occurs in batch. LIB/CCF also passes this information to exit 8 at a later step.

This exit is particularly effective when it passes information from programmer to control group.

The PARM List Offset is as follows: +0 is the address of the User Data Area

$CCFU002

Invoked before the creation of a Module Movement Record (MMR). Normally, any user data in the Module Tracking Record (MTR) that exit 0 supplies during logout is carried over to MMRs and JSRs for the member. However, the administrator can opt to have exit 2 change the user data during login processing. Exit 2 can supply up to 42 bytes of user data.

This data is placed on the module movement record and any job submission records that the login request creates. If $CCFU002 is not used and $CCFU000 supplies data during the logout request, then the existing user data is carried forward to the module movement record and job submission record.

The PARM List Offsets are as follows:

- +0—The address of the User Data Area
- +4—The address of the Module Name
- +8—The address of the Master File in which the member currently resides.

$CCFU003

Invoked after the data is entered and verified on the logout panel. The exit can examine or modify the panel data fields that are supplied to it. The exit can issue a message through ISPLINK SETMSG service. It can defer processing and redisplay the panel by returning a non-zero return code in register 15.

The PARM List Offsets are as follows:

- +0—The address of the Panel Data MODULE field
- +4—The address of the Panel Data CR ID field
- +8—The address of the Panel Data M/F SELECT field
- +12—The address of the CCFID
- +16—The address of the Prod Master (the length is 44)
- +20—The address of the Test Master (the length is 44)

$CCFU004

Invoked before the history create and update panels display. Can standardize information used for history member creation. When a programmer logs out a member that has no associated history member on the history master file, LIB/CCF assumes that the member was never logged out before and that it should create a new history member. LIB/CCF displays the History Create panel to prompt the programmer for information about the member. Subsequent logins of the same member display the History Update panel. Enter the **HISTORY** command from Option 5 (LOGIN) to modify the existing history information. The *type indicator* listed below refers to each of these functions respectively (CREATE, UPDATE, and MODIFY).

Before displaying the History Create panel or the History Update panel, LIB/CCF calls exit 4. This exit provides some or all of the member's history information. This lets the administrator enforce any site standards that exist. To suppress the display of the History Create or Update panels, the exit can issue the ISPF CONTROL NONDISPL dialog service.

The PARM List Offsets are as follows:

- n +0—The address of the Start of Panel Data Area
- n +4—The address of the Type of Indicator:
  - C (CREATE)
  - U (UPDATE)
  - M (MODIFY)
  - I (BATCH LOGIN)
  - O (BATCH LOGOUT)
- +8—The address of the Member Name (the length is 8)
- +12—The address of the The Prod Master (the length is 44)
- +16—The address of the CR ID (the length is 8)

PANEL DATA AREA (CREATE, MODIFY, and LOGOUT functions) are as follows:

- n The System field has a length of 20
- n The Description field has a length of 30
- n The Language field has a length of 8
- n Lines 1 through 16 of the abstract have a length of 61

PANEL DATA AREA (UPDATE and LOGIN functions) is as follows:

- Lines 1 through 19 of Description have a length of 61

$CCFU005

Invoked after the programmer fills in a history create or update panel. The exit can verify data supplied when history creation is complete. The exit can also examine, verify, override, or reject what the programmer filled in. The exit can modify the panel data and issue a message through ISPLINK SETMSG service. It can defer processing and redisplay the panel by passing a non-zero return code in register 15.

The PARM List Offsets are as follows:

- +0—The address of the Star of the Panel Data Area
- +4—The address of the Type of Indicator:
  - C (CREATE)
  - U (UPDATE)
  - M (MODIFY)
  - I (BATCH LOGIN)
  - O (BATCH LOGOUT)
- +8—The address of the Member Name (the length is 8)
- +12—The address of the Prod Master (the length is 44)
- +16—The address of the CR ID (the length is 8)

PANEL DATA AREA (CREATE, MODIFY, and LOGOUT functions) are as follows:

- The System field has a length of 20
- The Description field has a length of 30
- The Language field has a length of 8
- Lines 1 through 16 of the abstract have a length of 61

PANEL DATA AREA (UPDATE and LOGIN functions):

- n Lines 1 through 19 of the Description have a length of 61

$CCFU006

Invoked after the login function completes successfully. The exit can provide the opportunity for special reports or issue messages.

The PARM List Offsets are as follows:

- +0—The address of the Member (the length is 8)
- +4—The address of the Prod Master (the length is 44)
- +8—The address of the Test Master (the length is 44)
- +12—The address of the CR ID (the length is 8)
- +16—The address of the TYPEIND (the length is 1)
- +20—The address of the Date/Time (the length is 17)

Sets the TYPE INDICATOR to C'1' to indicate a rejected entry was processed.

$CCFU007

Invoked before performing the member lock function. The exit can supply additional data during the update to lock a member on the test master file when a programmer issues a login request to the control group. Each update control statement is passed to the exit before it is written to the update stream. The exit can modify, delete, or add to the update stream in the process.

When a programmer logs in a member and the control group is responsible for the actual login, LIB/CCF locks the member on the test master file from further updates. Before LIB/CCF locks the member, it passes the CA Librarian control statements (that perform the lock) one by one to exit 7 for modifications and updates.

LIB/CCF loads the exit once and calls it multiple times, once per control statement. You might want the exit to maintain information on previous calls by maintaining flags. However, because LIB/CCF does not notify the exit of EOF, the exit does not know when to free storage it acquired. Therefore, the exit should avail itself of the 80-bye work area that LIB/CCF supplies. LIB/CCF does not modify this area, so the exit can use it any way you want.

To return a control statement unchanged, the exit passes back a return code of 0. LIB/CCF calls the exit again if there are more statements to process.

Even though there can be more statements to process, the exit can notify LIB/CCF not to call it again by passing back a return code of 4.

To insert a statement after the one LIB/CCF just passed, the exit passes back a return code of 8. LIB/CCF calls the exit again to insert the statement. The exit should return 8s until the last statement to insert, when it should return a zero. LIB/CCF then continues by passing the next control statement.

When the exit is in insert mode, it should keep track of that fact by maintaining some sort of flag in the work area.

A return code of 12 tells LIB/CCF to delete or not use the control statement that it just passed to the exit.

The PARM List Offsets are as follows:

- +0—The address of the Data Record (the length is 80)
- +4—The address of the Test Master (the length is 44)
- +8—The address of the Modname (the length is 8)
- +12—The address of the Test Password (the length is 44)
- +16—The address of the Test MCD (the length is 4)
- +20—The address of the User ID (the length is 8)
- +24—The address of the Work Area (the length is 80)

The return codes are as follows:

- 0—NORMAL (the data is processed)
- 4—TERMINATE calling the exit (the data record is processed and the exit is not called again)
- 8—INSERT (the data record is processed and the exit is called again to supply a record to insert)
- 12—DELETE the record

$CCFU008

Invoked before processing the link-edit SYSLIN data or link-edit JCL member. The exit can examine and modify the linkage editor control statements during link-edit request processing. The data contained in the member is passed to the exit one record at a time. The exit can modify, delete, or add to the member during this process.

When a programmer initiates a production link request, he must specify one of two things. He can specify the name of a SYSLIN member on the production master file that contains linkage editor control statements.

LIB/CCF wraps JCL around it and submits the whole thing. Alternatively, the programmer can supply the name of a member, again on the production master file, that contains an entire job stream to submit for the link-edit.

In either case, before processing the SYSLIN or JCL member, LIB/CCF can pass it line by line to exit 8 for scrutiny and modification. Exit 8 can work with exit 1.

Keep in mind the different sequences of events that can occur. When LKED=BATCH or ONLINE, the programmer initiates the link-edit and sees it through. LIB/CCF calls exit 8 right after exit 1. In this situation, there is no reason to use both exits when exit 8 can do the job of both; that is, get information and use it to customize the link-edit SYSLIN or JCL member.

But when LKED=CNTL, a time gap separates the link-edit initiation (by the programmer) and its being carried out (by the control group). LIB/CCF calls exit 1 when the programmer initiates the production link request and exit 8 only when the control group processes the request. Therefore, there is a natural separation of function: Exit 1 to supply information from the programmer to the control group; exit 8 to use that information, supplied by LIB/CCF during the exit call, to customize the link-edit SYSLIN or JCL member.

In either case, before it submits the actual job stream, LIB/CCF then calls exit 9 to touch up the JCL that is wrapped around the linkage editor statements.

The PARM List Offsets are as follows:

- +0—The address of the Data Record (the length is 80)
- +4—The address of the User Data Area (the length is 56)
- +8—The address of the Work Area (the length is 80)

The return codes are as follows:

- 0—NORMAL (the data record is processed)
- 4—TERMINATE calling the exit (the data record is processed and the exit is not called again)
- 8—INSERT (the data record is processed and the exit is called again to supply a record to insert)
- 12—DELETE the record and call again if there are more records.

$CCFU009

Invoked during submit processing before writing the record to the reader. The exit can modify, delete, or add to the JCL stream during this process.

LIB/CCF calls exit 9 every time it is ready to submit a job stream to accomplish logouts, logins, or link-edits. Since the exit does not know what the job does, it must limit itself to modifying only the JCL itself and not the control statements.

The PARM List Offsets are as follows:

- +0—The address of the Data Record (the length is 80)
- +4—The address of the SYSUID (the length is 8)
- +8—The address of the LOGON Password (the length is 8)
- +12—The address of the Work Area (the length is 80)

$CCFU010

Invoked by Option 9 when the PROCESS, REJECT, MOVE, or COMPILE command is entered and the movement of the member is toward production (as indicated by a TO: PROD indicator in the first line of the Module Movement Record in Option 9). The exit can restrict these commands to a subset of the control group.

The PARM List Offsets are as follows:

- +0—The address of the Member (the length is 8)
- +4—The address of the Production Master (the length is 44)
- +8—The address of the Test Master (the length is 44)
- +12—The address of the CR ID (the length is 8)
- +16—The address of the SYSUID (the length is 8)
- +20—The address of the User Data Area (the length is 56)
- +24—The address of the Command Indicator (the length is 1; the value can be P, R, M, or C)

The return codes are as follows:

- 0—Continue processing (CCF processes the record)
- 8—Defer processing (The request is not processed and the panel redisplays; the exit should set a message for the user)

$CCFU012

Invoked any time JCL is tailored for submission (with the exception of print jobs). The exit can modify batch jobs and provide variables for this purpose. The exit is called twice; before file tailoring and again after file tailoring. The exit can provide additional variables for file tailoring.

The Exit Data Area contains zeros on the first call to the exit. Computer Associates suggests you store the address of your dynamic storage area (used for VDEFINEs) in the Exit Data Area. The second call to the exit should free that dynamic storage area and VDELETE all variables that were VDEFINEd during the first call.

The PARM List Offsets are as follows:

- +0—The address of the Function Indicator (the length is 1)
  - I (Login)
  - O (Logout)
  - R (Reject)
  - L (Link Edit or Bind)
  - T (Test)
- +4—The address of the CDFDINFODSECT
- +8—The address of the Change Request ID (the length is 8)
- +12—The address of the Member Name (the length is 8)
- +16—The address of the Language for the Module (the length is 8)

  Note: Applicable only to Login.
- +20—The address of the User Data Area (the length is 56)
- +24—The address of the Exit Data Area (the length is 4)

$CCFU013

Invoked after the SYSTEM LINK or SYSTEM BIND panel is filled in. The exit can modify the override skeleton name supplied, reject it, or supply a new one.

The PARM List Offsets are as follows:

- +0—The address of the OVERRIDE SKELETON (the length is 8)
- +4—The address of the CR ID (the length is 8)
- +8—The address of the LINK/BIND SYSIN MEMBER (the length is 8)
- +12—The address of the CCF ID (the length is 8)

- +16—The address of the Application Name (the length is 20)
- +20—The address of the Function Indicator (the length is 1):
  - L (LINK)
  - B (BIND)

$CCFU014

Invoked after a change request is closed from Option 7 (CLOSE). The exit can interface to Unicenter CA-Netman (see Chapter 12, “LIB/CCF-Unicenter CA-Netman Interface,” for details).

The PARM List Offsets are as follows:

- +0—The address of the CR ID (the length is 8)
- +4—The address of the CA Netman ID (the length is 16)
- +8—The address of the System Master File DD Name (the length is 8)
- +12—The address of the System Master File MCD base code (the length is 4)

$CCFU015

Invoked before a change request is opened using Option 1 (OPEN). The exit can supply a site-specified change request number.

The PARM List Offset is as follows: +0, the address of the CR ID (the numeric length is 6).

Note: CR ID must be returned a numeric value length of 6.

Sample User Exits

The sample exit load modules are in the CA Librarian target load library. The sample exit source modules are in the CAI.CAISRC file. The CALJJCL library member LUCCFASM contains JCL to assemble and link the source for the sample user exits using the batch CA Librarian program. Member LUCCFASL contains the JCL to assemble and link the source using LIB/AM.

$CCFU000, $CCFU001, and $CCFU002

These three exits are similar in that they can provide user data to place in the LIB/CCF system records. They provide sample exits that display a panel asking for the date and time the request should be done and then put this data in the user area. $CCFU000 also asks for the member name that is linked. $CCFU001 uses this information.

$CCFU003

Provides a sample exit that prevents certain master files from being selected for logout. In this example, these master files are quality assurance libraries that another exit uses. They are described later. You can easily modify the exit to check the CCFID and permit only certain programmers to use certain master files or certain member names.

$CCFU004

Provides a sample exit that fills in the CREATE panel with default information and illustrates how panel data can be passed to the next exit. The information in the description field is saved in the SHARED variable pool through VPUT.

$CCFU005

Provides a sample exit that retrieves the data saved in the description field by the previous exit through a VGET from the SHARED profile pool. It then compares it to the description field currently on the panel. If the user changed the description, it is changed back, an error message issued, and a return code issued to redisplay the panel.

$CCFU006

Provides a sample exit that sends a NOTIFICATION message indicating there is a pending MMR to a list of control group members. You must modify the exit to add the names to a table.

$CCFU007

Provides a sample exit that inserts a -PGMR card into the update stream so that the programmer name field contains the CCFID of the user requesting the login.

$CCFU008

Provides a sample exit that takes the member name being linked from the user data area (placed there by $CCFU001). The link-edit SYSLIN data or JCL member is scanned for the symbolic @MODNAME@. The name from the user data area is substituted. If some of your members are linked in the same way, this technique could set up a single skeleton link member to handle several different members.

$CCFU009

Provides a sample exit that substitutes the SYSUID for *UID and the RACF PASSWORD for *PSW. You can code the skeleton job card with NOTIFY=*UID and with USER=*UID,PASSWORD=*PSW for RACF.

$CCFU010

Provides a sample exit that permits only some members of the control group to process MMRs once they reach a specified Q/A library. You must modify the exit's tables to supply the library names and SYSUID names.

$CCFU014

Provides a sample exit to interface with Unicenter CA-Netman. See Chapter 12, "LIB/CCF-Unicenter CA-Netman Interface," for details.

$CCFU015

Provides a sample exit to display a panel requesting a user-specified change request number during Option 1 (OPEN) processing. You must specify leading zeros in the change request number.

Sample User Exit $CCFU007

The following is an example of user exit $CCFU007. The exit inserts a PGMR card with the user ID as a programmer name in the appropriate place in the job stream.

```
TITLE 'TSO CCF SAMPLE USER EXIT 7'
*
* THIS EXIT USES THE FIRST BYTE OF THE RE-ENTRANT WORKAREA
* AS A FLAG WHOSE VALUE IS PRESERVED ACROSS EXIT CALLS
*
*PARMS:  DATARECORD,     (LENGTH 80)    <-- R2
*        TEST MASTER DSN,(LENGTH 44)    <-- R3
*        MODNAME,        (LENGTH 8)     <-- R4
*        TEST PASSWORD,  (LENGTH 4)     <-- R5
*        TEST MCD,       (LENGTH 4)     <-- R6
*        USERID,         (LENGTH 8)     <-- R7
*        USER WORK AREA  (LENGTH 80)    <-- R8 (RE-ENTRANT)
 *        JECT
```

```
$CCFU007 CSECT
         SAVE  (14,12)          SAVE CALLERS REGS
         LR    R12,R15          GET THE BASE REGISTER
         USING $CCFU007,R12     ESTABLISH BASE REGISTER
         LM R2,R8,0(R1)         GET THE PARM VALUES
         CLI 0 (R8),C'1'        CHECK FLAG IN RE-ENTRANT WKAREA
         BE    INSERT           IF IN INSERT MODE
         CLC 0 (4,R2),SEL       IS THIS THE -SEL CARD?
         BNE   GOBACK0          NO, RETURN TO CALLER

         LA    R15,8            INDICATE WOULD LIKE TO INSERT
         MVI 0 (R8),C'1'        MARK WORKAREA TO SET INS MODE
         B     GOBACK           EXIT
INSERT   MVC 0 (5,R2),PGMR      MOVE IN THE PGMR CARD
         MVC   6(8,R2),0(R7)    MOVE IN THE USER ID AS PGMR
         LA    R15,4            DO NOT CALL EXIT AGAIN
         B     GOBACK           EXIT

GOBACK0  XR    R15,R15          SET RC=0
*
GOBACK   EQU *
         RETURN (14,12),RC=(15) RETURN TO CALLER
*
* LOCAL CONSTANT STORAGE
*
PGMR     DC    CL5'-PGMR'
SEL      DC    CL4'-SEL'
*

R0       EQU   0
R1       EQU   1
R2       EQU   2
R3       EQU   3
R4       EQU   4

R5       EQU   5
R6       EQU   6
R7       EQU   7
R8       EQU   8
R9       EQU   9
R10      EQU   10
R11      EQU   11

R12      EQU   12
R13      EQU   13
R14      EQU   14
R15      EQU   15
         END   $CCFU007
```

JCL Skeletons

The JCL skeletons distributed with LIB/CCF-ISPF(TSO) reside in the LIB/CCF skeleton library created when LIB/CCF is installed. This skeleton library must be part of the ISPF ISPSLIB concatenation in the TSO procedure or allocation CLIST your site uses. The skeletons are:

$CCFJ000

Skeleton JOB card for batch submissions.

$CCFJ001

Skeleton JCL for CA Librarian Utility Copy - Production to test.

$CCFJ002

Skeleton JCL for CA Librarian Utility Copy - Test to Production.

$CCFJ003

Skeleton JCL for CA Librarian Production Application System LKED.

$CCFJ004

Skeleton JCL for Special CA Librarian Copy - Test to Production.

$CCFJ005

Skeleton JCL for object copy - Test to Production.

$CCFJ006

Skeleton JCL for Special CA Librarian Copy - Production to Test.

$CCFJ007

Skeleton JCL for Production to Backup Load Copy.

$CCFJ008

Skeleton JCL for batch print request.

$CCFJ009

Skeleton used to write SYSIN data to JCL stream.

$CCFJ010

Skeleton JCL for Production Source Assembly.

$CCFJ011

Skeleton JCL for Production Source COBOL Compile.

$CCFJ012

Skeleton JCL for Production Source PLI Compile.

$CCFJT10

Skeleton JCL for Test Source Assembly.

$CCFJT11

Skeleton JCL for Test Source COBOL Compile.

$CCFJT12

Skeleton JCL for Test Source PLI Compile.

$CCFJ030

Skeleton JCL for DB2 for z/OS and OS/390 precompile and assembly. For multiple DB2 for z/OS and OS/390 subsystems, create multiple copies of the skeleton, with the name of each subsystem hard-coded in the &$CCFDB2 field.

$CCFJ031

Skeleton JCL for DB2 for z/OS and OS/390 precompile and COBOL compile. For multiple DB2 for z/OS and OS/390 subsystems, create multiple copies of the skeleton, with the name of each subsystem hard-coded in the &$CCFDB2 field.

$CCFJ032

Skeleton JCL for DB2 for z/OS and OS/390 precompile and PLI compile. For multiple DB2 for z/OS and OS/390 subsystems, create multiple copies of the skeleton, with the name of each subsystem hard-coded in the &$CCFDB2 field.

$CCFJ040

Skeleton JCL for a conditional assembly and link-edit.

$CCFJ100

Skeleton JCL step for Automatic Job Checkoff.

$CCFJ101

Skeleton JCL for Test/Production Compare Optio JCL skeleton.

$CCFJ102

Skeleton JCL step for Automatic Job Checkoff for LKED jobs.

$CCFJ103

Skeleton JCL for DB2 for z/OS and OS/390 bind.

$CCFJ200

Skeleton JCL for CA MetaCOBOL+.

$CCFJ500

Skeleton JCL for batch login.

$CCFJ501

Skeleton JCL for batch logout.

Review each of the JCL skeletons to determine which requires modification to conform to your site requirements. Your LIB/CCF site does not reference every skeleton. However, it is a good practice to check each one so that, if you later update your system tables to reference different skeletons, you do not need to go back to see if anything else must be changed.

Note: To avoid S806 abends due to load modules not found, add either a JOBLIB or STEPLIB DD statement to the skeletons where appropriate.

$CCFJ000

Skeleton job card for batch submissions. This skeleton is filled in with the information entered in Option 0.2 (JOBCARD). The symbolics in this skeleton are:

- &$CCFJC1—First line of job card
- &$CCFJC2—Second line of job card
- &$CCFJC3—Third line of job card
- &$CCFJC4—Fourth line of job card

$CCFJ001

Skeleton JCL for CA Librarian Utility Copy - production to test. Copies all archive levels of a module. The symbolics in this skeleton are:

- &PRODMCD—Production master MCD code
- &TESTMCD—Test master MCD code
- &TSTATUS—Status specified for the test master
- &MODULENM—Module name
- &PMAST—DSN of production master file
- &TMAST—DSN of test master file
- &AFOLIBR—CA Librarian batch program name

$CCFJ002

Skeleton JCL for CA Librarian Utility Copy - test to production. Copies all archive levels of a module. The symbolics in this skeleton are:

- &PRODMCD—Production master MCD code
- &TESTMCD—Test master MCD code
- &PSTATUS—Status specified for the production master
- &MODULENM—Module name
- &VERS—Current CA Librarian version on production
- &PRODPSWD—Member password on production master
- &TESTPSWD—Member password on test master

- &PMAST—DSN of production master file
- &TMAST—DSN of test master file
- &AFOLIBR—CA Librarian batch program name

$CCFJ003

Skeleton JCL for CA Librarian production application system link requests. The symbolics in this skeleton are:

- &OPTS—Link-edit parm options
- &LPOBJ—Production object library
- &LPLOAD—Production load library

$CCFJ004

Skeleton JCL for LIBRCOPY - test to production. Copies only the current archive level of a module. The symbolics in this skeleton are:

- &PRODMCD—Production master MCD code
- &TESTMCD—Test master MCD code
- &PSTATUS—Status specified for the production master
- &MODULENM—Module name
- &VERS—Current CA Librarian version on production master
- &PRODPSWD—Member password on production master
- &TESTPSWD—Member password on test master
- &PMAST—DSN of production master file
- &TMAST—DSN of test master file
- &AFOLIBR—CA Librarian batch program name

$CCFJ005

Skeleton JCL for copy of OBJECT - test to production. The symbolics in this skeleton are:

- &TOBJECT—Test object library
- &POBJECT—Production object library
- &MODULENM—Module name

$CCFJ006

Skeleton JCL for LIBRCOPY - production to test. Copies only the current archive level of a module. The symbolics in this skeleton are:

- &PRODMCD—Production master MCD code
- &TESTMCD—Test master MCD code

- &PSTATUS—Status specified for the production master
- &MODULENM—Module name
- &PMAST—DSN of production master file
- &TMAST—DSN of test master file
- &AFOLIBR—CA Librarian batch program name

$CCFJ007

Skeleton JCL for production to backup LOAD copy. The symbolics in this skeleton are:

- &BKLOAD—Production backup load library
- &LPLOAD—Production load library

$CCFJ008

Skeleton JCL for batch print request. The symbolics in this skeleton are:

- &SYSOUT—Sysout class to use
- &DEST—DEST to use
- &COPIES—Number of copies to use

$CCFJ009

Skeleton used for writing SYSIN data to JCL. The symbolic in this skeleton is:

- &JCLREC—SYSIN record

$CCFJ010

Skeleton JCL for production source assembly. The symbolics in this skeleton are:

- &MODULENM—Module name
- &PRODPSWD—Password on production master file
- &PRODMCD—Production master file MCD code
- &POBJECT—Production object library
- &PMAST—Production master file name
- &AFOLIBR—CA Librarian batch program name

$CCFJ011

Skeleton JCL for production source COBOL compile. The symbolics in this skeleton are:

- &MODULENM—Module name
- &PRODPSWD—Password on production master file
- &PRODMCD—Production master file MCD code

- &POBJECT—Production object library
- &PMAST—Production master file name
- &AFOLIBR—CA Librarian batch program name

$CCFJ012

Skeleton JCL for production source PL1 compile. The symbolics in this skeleton are:

- &MODULENM—Module name
- &PRODPSWD—Password on production master file
- &PRODMCD—Production master file MCD code
- &POBJECT—Production object library
- &PMAST—Production master file name
- &AFOLIBR—CA Librarian batch program name

$CCFJT10

Skeleton JCL for test source assembly. With the exception of &AFOLIBR and &MODULENM, only the extension variables (documented in Appendix A) are valid for this skeleton. The symbolics in this skeleton are:

- &MODULENM—Module name
- &IFMC—Test master file MCD code
- &IMFO—Test object library
- &IFMAST—Test master file name
- &AFOLIBR—CA Librarian batch program name

$CCFJT11

Skeleton JCL for test source COBOL compile. With the exception of &AFOLIBR and &MODULENM, only the extension variables (documented in Appendix A) are valid for this skeleton. The symbolics in this skeleton are:

- &MODULENM—Module name
- &IFMC—Test master file MCD code
- &IMFO—Test object library
- &IFMAST—Test master file name
- &AFOLIBR—CA Librarian batch program name

$CCFJT12

Skeleton JCL for test source PLI compile. With the exception of &AFOLIBR and &MODULENM, only the extension variables (documented in Appendix A) are valid for this skeleton. The symbolics in this skeleton are:

- &MODULENM—Module name
- &IFMC—Test master file MCD code
- &IMFO—Test object library
- &IFMAST—Test master file name
- &AFOLIBR—CA Librarian batch program name

$CCFJ030

Skeleton JCL for DB2 for z/OS and OS/390 precompile and assembly. For multiple DB2 for z/OS and OS/390 subsystems, create multiple copies of the skeleton, with the name of each subsystem hard-coded in the &$CCFDB2 field. The symbolics in this skeleton are:

- &AFOLIBR—Batch CA Librarian name
- &MODULENM—Member name
- &$CCFDB2—DB2 for z/OS and OS/390 subsystem name
- &ITT—Target library type
- &ITMCD—Target master file MCD
- &IHMCD—History master file MCD
- &ITMAST—Target master file
- &IHMAST—History master file
- &IMTO—z/OS and OS/390 target object library
- &IMTD—z/OS and OS/390 target DBRM library
- &WO—Change request number
- &CCFID—CCFID

$CCFJ031

Skeleton JCL for DB2 for z/OS and OS/390 precompile and COBOL compile. For multiple DB2 for z/OS and OS/390 subsystems, create multiple copies of the skeleton, with the name of each subsystem hard-coded in the &$CCFDB2 field. The symbolics in this skeleton are:

- &AFOLIBR—Batch CA Librarian name
- &MODULENM—Member name
- &$CCFDB2—DB2 for z/OS and OS/390 subsystem name

- &ITT—Target library type
- &ITMCD—Target master file MCD
- &IHMCD—History master file MCD
- &ITMAST—Target master file
- &IHMAST—History master file
- &IMTO—z/OS and OS/390 target object library
- &IMTD—z/OS and OS/390 target DBRM library
- &WO—Change request number
- &CCFID—CCFID

$CCFJ032

Skeleton JCL for DB2 for z/OS and OS/390 precompile and PL1 compile. For multiple DB2 for z/OS and OS/390 subsystems, create multiple copies of the skeleton, with the name of each subsystem hard-coded in the &$CCFDB2 field. The symbolics in this skeleton are:

- &AFOLIBR—Batch CA Librarian name
- &MODULENM—Member name
- &$CCFDB2—DB2 for z/OS and OS/390 subsystem name
- &ITT—Target library type
- &ITMCD—Target master file MCD
- &IHMCD—History master file MCD
- &ITMAST—Target master file
- &IHMAST—History master file
- &IMTO—z/OS and OS/390 target object library
- &IMTD—z/OS and OS/390 target DBRM library
- &WO—Change request number
- &CCFID—CCFID

$CCFJ040

Skeleton JCL to conditionally assemble and link edit at login time. If the movement is to production, the batch CA Librarian is executed with output exit LIBREXIT to insert Source to Load Audit Trail (SLAT) variables and perform the link-edit into the production load library. If the movement is not to production (that is, to a quality assurance library), LIB/AM assembles the member directly from the master file without performing the link-edit. The symbolics in this skeleton are:

- &AFOLIBR—Batch CA Librarian name
- &MODULENM—Member name
- &PRODPSWD—Production member's password
- &ITT—Target library type
- &ILOCK—Security lock status
- &ITMCD—Target master file MCD
- &ITMAST—Target master file
- &IMTO—z/OS and OS/390 target object library
- &IMTL—z/OS and OS/390 target load library

$CCFJ100

Skeleton JCL step for automatic job checkoff. This skeleton executes program $CCFB100, which dynamically allocates the CCF system master file based on the SYSMAST option of the $CCFGEN macro. The $CCFGEN macro creates a load module called $CCFCOMI. Therefore, this skeleton can require a STEPLIB DD statement for the load library that contains both $CCFB100 and $CCFCOMI. This skeleton has no symbolics.

$CCFJ101

Skeleton for Test/Production Compare option. This skeleton executes Comparator, which is site named at installation time. You must update this skeleton with the appropriate program name. The symbolics in this skeleton are:

- &MODULENM—Module name
- &PRODMCD—Production master file MCD code
- &PMAST—Production master file name
- &TESTMCD—Test master file MCD code
- &TMAST—Test master file name
- &LOLEVEL—Version date and time of production level logged out

$CCFJ102

Skeleton JCL step for automatic job checkoff for batch link-edits. This skeleton executes program $CCFB100, which dynamically allocates the CCF system master file based on the SYSMAST option of the $CCFGEN macro. The $CCFGEN macro creates a load module called $CCFCOMI. Therefore, this skeleton can require a STEPLIB DD statement for the load library that contains both $CCFB100 and $CCFCOMI. This skeleton has no symbolics.

$CCFJ103

Skeleton JCL for production DB2 for z/OS and OS/390 bind. The symbolics in this skeleton are:

- &IMTD—Target DBRM library
- &$CCFDB2—DB2 for z/OS and OS/390 subsystem name

$CCFJ200

Skeleton JCL for CA MetaCOBOL+. The symbolics in this skeleton are:

- &MODULENM—Module name
- &PRODPSWD—Password for production module
- &PRODMCD—Production master file MCD code
- &POBJECT—Production object library
- &PMAST—Production master file name
- &AFOLIBR—CA Librarian batch program name

$CCFJ201

Skeleton JCL for Unicenter CA-APCDOC for JCL procs. The symbolics in this skeleton are:

- &MODULENM—Module name
- &PRODPSWD—Password for production module
- &PRODMCD—Production master file MCD code
- &POBJECT—Production object library
- &PMAST—Production master file name
- &AFOLIBR—CA Librarian batch program name

$CCFJ500

Skeleton JCL for batch login. The NXT execution parameter can bypass the use of exits. See the “LIB/CCF Batch Utilities for CA Roscoe and ISPF(TSO)” chapter for more details on the NXT parameter. Otherwise, do not modify this skeleton.

$CCFJ501

Skeleton JCL for batch logout. Do not modify this skeleton.

Manipulating JCL Processing

This section describes how you can alter LIB/CCF JCL processing and sequences of events.

As supplied on the installation tape, the JCL skeletons perform certain functions in a certain order.

LIB/CCF JCL skeletons are ISPF skeleton definitions, stored in an ISPF skeleton library, and processed through the ISPF file tailoring services. You can change the skeletons by directly editing the skeleton library. They are interpreted during ISPF execution. LIB/CCF builds a job stream from any number of skeletons using the ISPF file tailoring services. ISPF individually processes each single piece and appends them together until a complete job stream is constructed when it is submitted.

Login through LIBRCOPY. First, it fetches skeleton $CCFJ004. This skeleton has four parts to it.

- The first part is an include for skeleton $CCFJ000, which is simply the job card.
- The second part is labeled step GETTEST, the first half of LIBRCOPY.
- The third part is labeled step PUTPROD, the second half of LIBRCOPY.
- The fourth part is labeled step DELTEST, to delete the member from the test master file.

By definition, this skeleton has symbolics that must be replaced by actual values before it can be used as JCL. The most important symbolics are &TMAST, &PMAST, and &MODULENM, which represent the test master file, the production master file, and the member name, respectively.

After $CCFJ004 is processed, LIB/CCF accesses the JCL skeleton for the compile. LIB/CCF determines the appropriate JCL compile skeleton by obtaining the language type for the member from the History Master file. It then uses this to search the LIB/CCF Language Definition Table. Look at skeleton $CCFJ010, which does assemblies.

There are two parts.

- The first selects the member from the production master file and writes it to OSJOB.
- The second part assembles the member.

The significant symbolics here are &PMAST, &MODULENM, and &POBJECT. &POBJECT is the production object library.

LIB/CCF appends $CCFJ010 to the job stream that was constructed thus far, then appends the JSR Checkoff step to the very end - skeleton $CCFJ100. This is the job stream that is submitted.

Once you understand the mechanics of how LIB/CCF builds JCL streams, it is clear that by manipulating the contents of the individual skeletons, job streams can be built in any way. The following sections examine two common scenarios.

Compile from the Test Master File

During login, some sites want the member compiled before it is moved from test to production. This means compiling from the test master file. You can do this with some minor changes to the JCL skeleton $CCFJ004 and the compile skeleton for the appropriate language type. In our example, the assembler skeleton, $CCFJ010, is used.

Remember that skeleton $CCFJ004 has four logical parts to it. Leave the first part intact (the include for the JOB card). Move the remaining three parts (GETTEST, PUTPROD, and DELTEST) to the end of skeleton $CCFJ010, following STEP2, the assembly step. The first part of $CCFJ010 selects the member from the production master file. Change the symbolic &PMAST to &TMAST to select the member from the test master file instead.

You need to make these changes for each language type that requires this processing.

Compile and Link at Login

Because many programs consist of only one member, you might often want to compile and link at login instead of in separate steps. This saves the programmer the extra step of submitting a System Link Request. You can do this with modifications to the compile JCL skeletons for each language type.

Using an assembler example, append an IEWL step to the assembly step in skeleton $CCFJ010. The skeleton now has three steps: extract (STEP1), assembly (STEP2), and link. In the assembly step, change &POBJECT to &IMTO (the production object library) on the SYSPUNCH DD. In the link step, specify &IMTO for the DSN on the OBJECT DD. Then, on the SYSLIB and SYSLMOD DDs, specify &IMTL (the production load library) for the DSN.

The z/OS and OS/390 OBJECT LIBRARY specified in LCDF (Option 12.11) for the production master file is substituted in the JCL skeleton for the &IMTO variable. Similarly, the z/OS and OS/390 LOAD LIBRARY is substituted for the &IMTL variable.

Alternatively, you can eliminate the use of a permanent object library since the assembly and link are performed in the same job. If so, instead of using the &IMTO variable for the object library, specify a temporary data set for the SYSPUNCH DD in the assembly step. Refer to that data set as the OBJECT DD in the link step.

Keep in mind that since the link-edit is included in the login step, the programmer can no longer execute the LIB/CCF System Link Request (Option 6). As a result, the programmer is not prompted for link-edit options like NCAL, LIST, and so on, at the time the link-edit is submitted. They must be hardcoded into the skeleton.

Note the following:

- For additional information on variables, see Appendix A.
- JCL skeleton $CCFJ040 conditionally performs an assembly and link-edit based on whether the module is moved into a Q/A or production library. It also uses LIB/AM and the output exit LIBREXIT.

Link-Edit Considerations

Several Option 12.0 (PROFILE) keywords affect the way link-edits are performed in LIB/CCF. They are CLINK, LKED, and LNKNTEGRTY. See the section titled System Profile Table in Chapter 3, “Administrator Functions in LIB/CCF,” for details on each keyword.

From Option 6 (LINK), the programmer is prompted for the LIB/CCF Change Request ID and the function to perform (a link or DB2 for z/OS and OS/390 bind). For details on the DB2 for z/OS and OS/390 bind process, see the section titled Initiating a Production Bind Request in the *LIB/CCF User Guide*. The panel defaults to L for link. The programmer sees a list of production master files and their associated production load libraries, as shown in the following sample. The programmer must then enter the number of the pair that defines the master file containing the associated linkage editor input and the load library where the modules should be linked.

```
THU mm/dd/yy ----------------- SYSTEM LINK EDIT ---------------- 15.42.44

COMMAND ===>

PRODUCTION MASTER FILE  ===> 001                REQUEST ID ===> WO000004

LKED SYSIN MEMBER ===>              OR     LKED JCL MEMBER ===>
OVERRIDE SKELETON ===>

AVAILABLE FILES: CHAIN DESC./CHAIN NUMBER/PRODUCTION/LOAD:

001) PAYROLL CHAIN                                    (00001)
         LIBR.PROD.MAST
             LIBR.PROD.LOAD

002) ACCOUNTING CHAIN                                 (00002)
         LIBR.PROD.MAST2
             LIBR.PROD.LOAD2
*** END OF LIST ***
```

The programmer can enter:

- Just a LKED SYSIN MEMBER, which uses the default skeleton if batch link-edits are performed,
- Just a LKED JCL MEMBER, which provides a complete JCL stream,
- Or both a LKED JCL member and an override skeleton that are used in place of the default skeleton.

LKED SYSIN MEMBER

This keyword is:

- Mutually exclusive with LKED JCL MEMBER.
- Must reside on the production master file selected in Option 6.
- Must contain the required link-edit INCLUDE statements that reference a ddname of OBJECT and a link-edit NAME statement specifying the REPLACE(R) option. Other link-edit control statements are optional as required.
- For batch link-edits, this member is used as SYSLIN to the default link-edit skeleton ($CCFJ003) or, if specified, the override link-edit skeleton.
- For online link-edits, this member is used as input to an execution of IEWL through a TSO TSOEXEC CALL.
- Specifying this field displays the Linkage Editor Options Selection panel. To choose an option, place an **X** next to it on the display. Selected options are inserted into the PARM= field of the EXEC statement in the JCL stream that is submitted.
- Automatic load module backup (if specified in LCDF) is based on load module names on the link-edit NAME statements.

LKED JCL MEMBER

This keyword is:

- Mutually exclusive with LKED SYSIN member and override skeleton.
- Must reside on production master file selected.
- Must be a complete link-edit JCL stream.
- LIB/CCF submits as is for execution.
- Job name is derived from the job card included in this member.
- Automatic load module backup is not performed unless a step is inserted into the JCL stream to accomplish it.

OVERRIDE SKELETON

This keyword is:

- Mutually exclusive with LKED JCL MEMBER.
- Must reside in the ISPF skeleton library allocated to ISPSLIB.

- Requires you supply a LKED SYSIN MEMBER.
- Specifies a skeleton to use in place of the default link-edit skeleton ($CCFJ003).
- Automatic load module backup (if specified in LCDF) is based on load module names on the link-edit NAME statements.

Note the following:

- If online link-editing is specified as an installation option (Option 12.0 specifies LKED ONLINE), only the LKED SYSIN MEMBER field appears on this panel.
- Computer Associates provides sample user exits to allow using a member name variable (@MODNAME@) in place of the actual member name in the LKED SYSIN MEMBER and LKED JCL MEMBER. See the section titled Sample User Exits earlier in this chapter for details on exits $CCFU001 and $CCFU008.

Customizing Panels and Reports

LIB/CCF is distributed with an Advantage CA-Earl (Easy Access Report Language) interface. Advantage CA-Earl is a report generation system that uses a high-level programming language to produce customized reports. The LIB/CCF-Advantage CA-Earl interface provides access to information LIB/CCF collected and stored on the System Master file. See the “LIB/CCF-Advantage CA-Earl Interface” chapter for details.

In addition to the LIB/CCF-Advantage CA-Earl interface, Computer Associates provides a macro ($CCFCOMT) to allow modification to LIB/CCF panel and report titles. After assembling and link editing the macro with the overrides, *both* the panel titles and the report headings are affected (that is, it is not necessary to modify the actual LIB/CCF panels). Reports printed online from LIB/CCF and by the batch change request status report program ($CCFB101) reflect titles the macro changed.

The following table describes the panel titles that the macro can modify (other titles on the panels are not modifiable).

Option	Panel	Title Field	Macro Label	Max Length
1-OPEN	$CCFPN01	REQUESTOR PHONE MANAGER	TITLE1	11
2-ASSIGN	$CCFPN02	DEPARTMENT COORDINATOR PHONE	TITLE2	11
3-DISPLAY	$CCFPN03		TITLE3	11
			TITLE4	11
			TITLE5	11
			TITLE6	11
2-ASSIGN 3-DISPLAY	$CCFPN02 $CCFPN03	MGR COMMENT	TITLE7	11
7-CLOSE	$CCFPN07	ANALYSIS	TITLE8	8

Option	Panel	Title Field	Macro Label	Max Length
7-CLOSE	$CCFPN19	ROUTINES AFFECTED	TITLE9 TITLE10	11 11
7-CLOSE	$CCFPN20	SYSTEMS TESTS MODIFIED ON SYSTEMS TESTS VERIFIED ON SYSTEMS TESTS ADDED OR MOD NEW USER INPUT NEW USER OUTPUT DOCUMENTATION CHANGES	TITLE11 TITLE12 TITLE13 TITLE14 TITLE15 TITLE16	25 25 26 14 15 21
4-LOGOUT 5-LOGIN	$CCFPN14 $CCFPN15	SYSTEM DESCRIPTION ABSTRACT	TITLE17 TITLE18 TITLE19	11 11 11
5-LOGIN	$CCFPN15	DESCRIPTION OF CHANGES	TITLE20 TITLE21	11 11

The $CCFCOMT macro, contained in the CA Librarian target macro library (CAI.CAIMAC), must be assembled and linked to specify title overrides. Computer Associates provides sample JCL in member LJCCFCMT on the CA Librarian CALJJCL library.

Note: The title appears on the panels and in the reports exactly as entered in the override. It can be right or left justified, contain embedded blanks, and be lower or upper case. You must delimit titles with apostrophes. Additionally, you can modify the following panels as indicated following:

Panel	Description	Help Panel
$CCFPN00	Primary LIB/CCF panel. This is the panel where you select the LIB/CCF functions. You can modify this panel to remove options that your site does not use.	$CCFH005
$CCFPN08	This is the Option 8 panel to select a report. You can modify this panel to remove reports or add additional reports.	$CCFH080
$CCFPN18	This is the Option 6 panel for the LKED options specification. You can modify this panel to specify different default options.	None

Execution

You execute LIB/CCF by selecting Option C on the CA Librarian selection menu (Option L from the main ISPF menu).

PF Keys and the Jump Command

You can display and modify PF key definitions for LIB/CCF the ISPF KEYS command from any LIB/CCF panel or Option 0.1. You can modify the PF keys settings. They affect the current and subsequent LIB/CCF sessions for the user. However, changing the PF key settings while in LIB/CCF does **not** affect the PF key settings for ISPF. LIB/CCF maintains its own PF key settings in the user's ISPF profile. Conversely, changing PF key settings while in ISPF does not affect LIB/CCF PF key settings.

The default settings are:

- PF1 and PF13 (- HELP)
- PF2 and PF14 (- SPLIT)
- PF3 and PF15 (- END)
- PF4 and PF16 (- RETURN)
- PF5 and PF17 (- RFIND)
- PF6 and PF18 (- RCHANGE)
- PF7 and PF19 (- UP)
- PF8 and PF20 (- DOWN)
- PF9 and PF21 (- SWAP)
- PF10 and PF22 (- LEFT)
- PF11 and PF23 (- RIGHT)
- PF12 and PF24 (- CURSOR)

LIB/CCF supports a jump command that enables a direct jump to a specified option.

This command has the following format:

```
=option[.option]
```

For example, you could enter =3 or =8.5.

Job Card Specification

The JCL JOB statement skeleton is used when LIB/CCF jobs are submitted for batch execution. Appropriately modify the job statement using Option 0.2 to reflect the correct accounting information, programmer name, job and message class, and so on.

The JOB statement can consist of up to four lines and must begin on the first line. Any trailing unused lines are ignored.

System Error Processing

In ISPF(TSO), there are three major types of error.

- The first is an ISPF dialogue error. ISPF displays a panel pinpointing where and why the error occurred.
- The second occurs when LIB/CCF abends with a code of hex 800 or decimal 2048. Should this abend occur, error messages are stored in a sequential data set named *prefix.userid*.CCF.MSGS, where *prefix* is your TSO user prefix. If PROFILE NOPREFIX is set, the data set name is simply *userid*.CCF.MSGS. View this data set immediately after notification of an error, as LIB/CCF reuses the data set for recording of subsequent messages.
- The third error is a file allocation error. In addition to the various LIB/CCF master files, LIB/CCF uses a number of temporary data sets for work files that it first allocates and then deletes on completion. Occasionally an error in LIB/CCF causes subsequent LIB/CCF executions to fail because a work file was not deleted. You can press PF1 to get more information about the data set that is failing allocation. Leaving and re-entering LIB/CCF usually clears the problem. In some cases, you might have to delete the file in question yourself.

In general, when you press PF1 after an error occurs, ISPF displays more information. If the information is in multiple screens, press PF1 again. For LIB/CCF processing errors, you can find the messages in *prefix.userid*.CCF.MSGS.

Chapter 14: Customizing Your System: LIB/CCF-ISPF(VM/ESA)

You can tailor LIB/CCF-ISPF(VM/ESA) to your site's needs in a number of ways over and above the options available to the system administrator on the LIB/CCF Table Processors. A number of exit points and sample exits are available. You need to tailor JCL skeletons to your site's requirements. You can also change panels and reports.

This section contains the following topics:

User Exits (see page 263)
JCL and EXEC Skeletons (see page 275)
Link-Edit Considerations (see page 290)
Customizing Panels and Reports (see page 292)
Execution (see page 293)
PF Keys and the Jump Command (see page 294)
Job Card Specification (see page 295)
System Error Processing (see page 296)

User Exits

LIB/CCF-ISPF (VM/ESA) lets you supply exit programs that are invoked at defined points to perform site-dependent processing. Computer Associates provides sample exits. Write exits for ISPF(VM/ESA) in Assembler language. They should be re-entrant and re-useable.

To make the exits available to LIB/CCF-ISPF(VM/ESA), they must be placed in an OS-simulated load library, that is, a VM/ESA file whose filetype is LOADLIB.

Follow these steps to create an exit.

1. Assemble the exit program.
2. Add the resulting text file to an OS-simulated text library (that is, a VM/ESA file whose filetype is TXTLIB). Use the VM/ESA TXTLIB command to manipulate the textlib.

   For example, to add $CCFU000 TEXT to EXITS TXTLIB, use the following command:

   TXTLIB ADD EXITS $CCFU000

   EXITS TXTLIB and $CCFU000 TEXT are on accessed read/write disks.

3. Issue the VM/ESA LKED command to create a relocatable module and store it in an OS-simulated load library. If the exit uses the ISPF services, you must make the txtlib that contains the IBM ISPLINK TEXT available to the LKED command.

   For example, to link edit the $CCFU000 relocatable module, use the following commands:

   ```
   FILEDEF SYSLIB DISK EXITS TXTLIB A
   FILEDEF SYSLIB2 DISK ISPLINK TXTLIB A
   FILEDEF SYSLMOD DISK EXITS LOADLIB A
   LKED SYSLIN ( LIBE SYSLMOD XREF MAP
   ```

   ISPLINK TXTLIB A contains ISPLINK TEXT. EXITS TXTLIB A contains $CCFU000 TEXT. EXITS LOADLIB A is the OS-simulated load library.

   Also, the LKED command refers to SYSLIN TEXT, which contains:

   ```
   INCLUDE SYSLIB($CCFU000)
    INCLUDE SYSLIB2(ISPLINK)
    ENTRY $CCFU000
   NAME $CCFU000(R)
   ```

   Records must begin in column 2.

4. After creating the load module, copy the load library onto an ISPF system disk that all ISPF users share. You must then modify the ISPF startup exec to chain EXITS LOADLIB into the ISPLLIB concatenation.

5. Finally, if the exit displays one or more ISPF panels, add the panels to a private or system ISPF panel library and update the ISPPLIB concatenation accordingly.

LIB/CCF executes the user exits if the EXITn= keyword of Option 12.0 (PROFILE) indicates it should. LIB/CCF can load the exits or they can link with LIB/CCF as specified in the $CCFGEN macro at installation. Computer Associates provides sample exit programs on the installation library in both source and object form. The source files for the sample programs are named $CCSU0*nn*, where *nn* is the exit number. The object modules for the sample programs are named $CCFU0*nn*, where *nn* is the exit number. The exits must be re-enterable and re-useable.

If the exit acquires storage, then it must release it before returning to LIB/CCF. However, if your exit needs to access and update data that must be preserved across exit calls, it should take advantage of the work area supplied for exits 7, 8, and 9. LIB/CCF supplies a clean work area the first time it calls any one of the four exits. For subsequent calls, LIB/CCF passes the address of the work area. LIB/CCF itself does not modify the work area. Therefore, your exit can use the work area in any way you want it to.

It is the responsibility of the exit program to SAVE and RESTORE the ISPF display environment when the exit displays a panel. Failure to do so can cause dialog errors during the display of CCF panels. The ISPF dialog services that preserve the original display environment are CONTROL,DISPLAY,SAVE and CONTROL,DISPLAY,RESTORE. See sample exits $CCSU000, $CCSU001, or $CCSU002 for an example of this processing.

Code all LIB/CCF exit routines using standard OS linkage conventions.

On entry, the contents of the registers are as follows:

- 13—Address of an 18 fullword save area
- 14—Return address
- 15—Entry address
- 1—Address of a parameter list

On return:

- 15—Return code

The details of the parameter lists passed to each exit routine and the meaning of the return codes each exit uses are summarized following.

$CCFU000

Invoked before the creation of a Module Tracking Record (MTR). The exit can supply up to 42 bytes of data to place on the MTR. Unless changed later, LIB/CCF duplicates this data on any subsequent MMRs and login JSRs for this member.

The PARM List Offset is +0, the address of the User Data Area.

$CCFU001

Invoked before the creation of a System Link Record (SLR). When the programmer initiates a production link request, LIB/CCF calls exit 1, which can supply up to 56 bytes of data to place on the SLR LIB/CCF creates. LIB/CCF duplicates this information on any subsequent JSRs if the link-edit is to occur in batch. LIB/CCF also passes this information to exit 8 at a later step.

This exit is particularly effective when it passes information from programmer to control group.

The PARM List Offset is +0, the address of the User Data Area.

$CCFU002

Invoked before the creation of a Module Movement Record (MMR). Normally, any user data in the MTR supplied by exit 0 during logout is carried over to MMRs and JSRs for the member. However, the administrator can have exit 2 change the user data during login processing. Exit 2 can supply up to 42 bytes of user data.

This data is placed on the module movement record and any job submission records that the login request created. If $CCFU002 is not used and $CCFU000 supplies the data during the logout request, then the existing user data is carried forward to the module movement record and job submission record.

The PARM List Offsets are as follows:

- +0—The address of the User Data Area
- +4—The address of the Module Name
- +8—The address of the Master File in which the member currently resides

$CCFU003

Invoked after the data is entered and verified on the logout panel. The exit can verify data supplied in the logout panel. The exit can examine or modify the panel data fields that are supplied to it. The exit can issue a message through ISPLINK SETMSG service. It can also defer processing and redisplay the panel by returning a non-zero return code in register 15.

The PARM List Offsets are as follows:

- +0—The address of the Panel Data MODULE field
- +4—The address of the Panel Data CR ID field
- +8—The address of the Panel Data M/F SELECT field
- +12—The address of the CCFID
- +16—The address of the PROD Master (the length is 44)
- +20—The address of the TEST Master (the length is 44)

$CCFU004

Invoked before the history create and update panels display. The exit can standardize information used for history member creation. When a programmer logs out a member that has no associated history member on the history master file, LIB/CCF assumes that the member was never logged out before and that it should create a new history member. LIB/CCF displays the History Create panel to prompt the programmer for information about the member. Subsequent logins of the same member display the History Update panel. Entering the HISTORY command from Option 5 (LOGIN) allows modification of the existing history information. The type indicator listed below refers to each of these functions respectively (CREATE, UPDATE, and MODIFY).

Before displaying the History Create panel or the History Update panel, LIB/CCF calls exit 4. This exit can provide some or all of the member's history information. This lets the administrator enforce any site standards that exist.

To suppress the display of the History Create and Update panels, the exit can issue the ISPF CONTROL NODISPL dialog service.

The PARM List Offsets are as follows:

- +0—The address of the start of the Panel Data Area
- +4—The address of the Type Indicator:
  - C (CREATE)
  - U (UPDATE)
  - M (MODIFY)
- +8—The address of the Member Name (the length is 8)
- +12—The address of the Prod Master (the length is 44)
- +16—The address of the CR ID (the length is 8)

PANEL DATA AREA (CREATE and MODIFY functions) are as follows:

- The System field has a length of 20
- The Description field has a length of 30
- The Language field has a length of 8
- Lines 1 through 16 of the abstract have a length of 61

PANEL DATA AREA (UPDATE function) is as follows:

- Lines 1 through 19 of the description have a length of 61

$CCFU005

Invoked after the programmer fills in the history create or update panel. The exit can verify data supplied when history member creation/update is complete. The exit can examine, verify, override, or reject what the programmer filled in. The exit can modify the panel data and issue a message through ISPLINK SETMSG service. Passing a non- zero return code in register 15 defers processing and redisplays the panel.

The PARM List Offsets are as follows:

- +0—The address of the start of the Panel Data Area
- +4—The address of the Type Indicator:
  - C (CREATE)
  - U (UPDATE)
  - M (MODIFY)
- +8—The address of the Module Name (the length is 8)
- +12—The address of the Prod Master (the length is 44)
- +16—The address of the CR ID (the length is 8)

PANEL DATA AREA (CREATE and MODIFY functions):

- The System field has a length of 20
- The Description field has a length of 30
- The Language field has a length of 8
- Lines 1 through 16 of the abstract have a length of 61

PANEL DATA AREA (UPDATE function):

- Lines 1 through 19 of the description have a length of 61

$CCFU006

Invoked after the login function is successfully complete. This exit can provide the opportunity for special reports.

The PARM List Offsets are as follows:

- +0—The address of the Member (the length is 8)
- +4—The address of the Prod Master (the length is 44)
- +8—The address of the Test Master (the length is 44)
- +12—The address of the CR ID (the length is 8)
- +16—The address of the TYPEIND (the length is 1)
- +20—The address of the Date/Time (the length is 17)

Sets the TYPE INDICATOR to C'1' to indicate a rejected entry was processed.

$CCFU007

Invoked before performing the member lock function. The exit can supply additional data during the update to lock a member on the test master file when the programmer issues a login request to the control group. Each update control statement is passed to the exit before being written to the update stream. The exit can modify, delete, or add to the update stream in the process.

When a programmer logs in a member and the control group is responsible for the actual login, LIB/CCF locks the member on the test master file from further updates. Before LIB/CCF executes online CA Librarian update control cards that lock the member, it passes them one by one to exit 7 for modifications and updates.

LIB/CCF loads the exit once and calls it multiple times, once per control statement. You can have the exit maintain information on previous calls by maintaining flags. However, because LIB/CCF does not notify the exit of EOF, the exit does not know when to free storage it acquired. Therefore, the exit should use the 80-byte work area that LIB/CCF supplies. LIB/CCF does not modify this area, so the exit can use it any way you want.

To return a control statement unchanged, the exit passes back a return code of 0. LIB/CCF calls the exit again if there are more statements to process.

Even though there might be more statements to process, the exit can notify LIB/CCF not to call it again by returning a return code of 4.

To insert a statement after the one LIB/CCF just passed, the exit passes back a return code of 8. LIB/CCF calls the exit again to get the statement to insert. The exit should return 8s until the last statement to insert, for which it should return a zero. LIB/CCF then continues by passing the next control statement.

When the exit is in insert mode, it should keep track of that fact by maintaining some sort of flag in the work area.

A return code of 12 tells LIB/CCF to delete or not use the control statement that it just passed to the exit.

The PARM List Offsets are as follows:

- +0—The address of the Data Record (the length is 80)
- +4—The address of the Test Master (the length is 44)
- +8—The address of the Modname (the length is 8)
- +12—The address of the Test Password (the length is 4)
- +16—The address of the Test MCD (the length is 4)
- +20—The address of the User ID (the length is 8)
- +24—The address of the Work Area (the length is 80)

Return Codes are as follows:

- 0—NORMAL return code (the data record is processed)
- 4—TERMINATE calling the exit (the data record is processed and the exit is not called again)
- 8—INSERT (the data record is processed and the exit is called again to supply a record to insert)
- 12—DELETE the record

$CCFU008

Invoked before processing the link-edit SYSLIN data or link-edit JCL member. The exit can examine and modify the linkage editor control statements during link processing. The data contained in the member is passed to the exit one record at a time. The exit can modify, delete, or add to the member during this process.

When a programmer initiates a production link request, he must specify one of two things.

- He can specify the name of a SYSLIN member on the production master file that contains linkage editor control statements. LIB/CCF wraps JCL around it and submits the whole thing.
- Alternatively, the programmer can supply the name of a member, again on the production master file, that contains an entire job stream to submit for the link-edit.

In either case, before processing the SYSLIN or JCL member, LIB/CCF can pass it line by line to exit 8 for scrutiny and modification. Exit 8 might not work with exit 1.

Keep in mind the different sequences of events that can occur. When LKED=BATCH, the programmer initiates the link-edit and sees it through.

LIB/CCF calls exit 8 right after exit 1. In this situation, there is no reason to use both exits when exit 8 can do the job of both; that is, get information and use it to customize the link-edit SYSLIN or JCL member.

But when LKED=CNTL, a time gap separates the link-edit initiation (by the programmer) and its being carried out (by the control group). LIB/CCF calls exit 1 when the programmer initiates the production link request and exit 8 only when the control group processes the request. Therefore, there is a natural separation of function: Exit 1 to supply information from the programmer to the control group; exit 8 to use that information that LIB/CCF supplied during the exit call to customize the link-edit SYSLIN or JCL member.

In either case, before it submits the actual job stream, LIB/CCF then calls exit 9 to touch up the JCL that is wrapped around the linkage editor statements.

The PARM List Offsets are as follows:

- +0—The address of the Data Record (the length is 80)
- +4—The address of the User Data Area (the length is 56)
- +8—The address of the Work Area (the length is 80)

Return Codes are as follows:

- 0—NORMAL (the data record is processed)
- 4—TERMINATE calling the exit (the data record is processed and the exit is not called again)
- 8—INSERT (the data record is processed and the exit is called again to supply a record to insert)
- 12—DELETE the record and call again if there are more records

$CCFU009

Invoked during submit processing, before writing the record to the reader. The exit can modify, delete, or add to the JCL stream during this process.

LIB/CCF calls exit 9 every time it is ready to submit a job stream to accomplish logouts, logins, or link edits. Since the exit does not know what the job does, it must limit itself to modifying only the JCL itself and not the control statements.

The PARM List Offsets are as follows:

- +0—The address of the Data Record (the length is 80)
- +4—The address of the SYSUID (the length is 8)
- +8—The address of the LOGON PASSWORD (the length is 8)
- +12—The address of the Work Area (the length is 80)

Return Codes are as follows:

- 0—NORMAL (the data record is processed)
- 4—TERMINATE calling the exit (the data record is processed and the exit is not called again)
- 8—INSERT (the data record is processed and the exit is called again to supply a record to insert)
- 12—DELETE the record and call again if more records

$CCFU010

Invoked by Option 9 when the PROCESS, REJECT, MOVE, or COMPILE command is entered and the movement of the member is toward production (as indicated by a TO: PROD indicator in the first line of the Module Movement Record in Option 9). The exit can restrict these commands to a subset of the control group.

The PARM List Offsets are as follows:

- +0—The address of the Member (the length is 8)
- +4—The address of the Production Master (the length is 44)
- +8—The address of the Test Master (the length is 44)
- +12—The address of the CR ID (the length is 8)
- +16—The address of the SYSUID (the length is 8)
- +20—The address of the User Data Area (the length is 56)
- +24—The address of the Command Indicator (the length is 1; the value can be P, R, M, or C)

Return Codes are as follows:

- 0—Continue processing. CCF processes the record.
- 8—Defer processing. The request is not processed and the panel redisplays. The exit should set a message for the user in this case.

$CCFU012

Invoked any time JCL is tailored for submission (with the exception of print jobs). The exit can modify the JCL. The exit is called twice; before file tailoring and again after file tailoring. The exit can provide additional variables for file tailoring.

The Exit Data Area contains zeros on the first call to the exit. Store the address of your dynamic storage area (used for VDEFINEs) in the Exit Data Area. The second call to the exit should free that dynamic storage area and VDELETE all variables that were VDEFINEd during the first call.

The PARM List Offsets are as follows:

- +0—The address of the Function Indicator (the length is 1):
  - I (Login)
  - O (Logout)
  - R (Reject)
  - L (Link Edit or Bind)
- +4—The address of the Variable DSECT (the length is 8)
- +8—The address of the Change Request ID (the length is 8)
- +12—The address of the Member Name (the length is 8)
- +16—The address of the Language for Module (the length is 8)

  Note: Applicable only to Login.
- +20—The address of the User Data Area (the length is 56)
- +24—The address of the Exit Data Area (the length is 4)

$CCFU013

Invoked after the SYSTEM LINK or SYSTEM BIND panel is filled in. The exit can control the use of the link or bind override skeleton from Option 6 (LINK). The exit can modify the OVERRIDE SKELETON name supplied, reject it, or supply a new one.

The PARM List Offsets are as follows:

- +0—The address of the OVERRIDE SKELETON (the length is 8)
- +4—The address of the CR ID (the length is 8)
- +8—The address of the LINK/BIND SYSIN MEMBER (the length is 8)
- +12—The address of the CCF ID (the length is 8)

- +16—The address of the Application Name (the length is 20)
- +20—The address of the Function Indicator (the length is 1):
  - L (Link)
  - B (Bind)

$CCFU015

Invoked before a change request is opened using Option 1 (OPEN). The exit can supply a site-specified change request number.

The PARM List Offset is +0, the address of the CR ID (the length is 6).

Sample User Exits

$CCFU000, $CCFU001, and $CCFU002

These three exits are similar in that they can provide user data that is placed in the LIB/CCF system records. They provide sample exits that display a panel asking for the date and time the request should be done. They then put this data in the user area. $CCFU000 also asks for the name of the member to link. $CCFU001 uses this information.

$CCFU003

Provides a sample exit that prevents certain master files from being selected for logout. In this example, these master files are quality assurance libraries that another exit uses. You can easily modify this exit to check the CCFID and permit only certain programmers to use certain master files or certain member names.

$CCFU004

Provides a sample exit that fills in the panel with default information and illustrates how panel data can be passed to the next exit. The information in the description field is saved in the SHARED variable pool through VPUT.

$CCFU005

Provides a sample exit that retrieves the data saved in the description field by the previous exit through a VGET from the SHARED profile pool. It then compares it to the description field currently on the panel. If the user changed the description, it is changed back, an error message is issued, and a return code is issued to redisplay the panel.

$CCFU007

Provides a sample exit that inserts a -PGMR card into the update stream so that the programmer name field contains the CCFID of the user requesting the login.

$CCFU008

Provides a sample exit that takes the member name being linked from the user data area (placed there by $CCFU001). The link-edit SYSLIN data or JCL member is scanned for the symbolic @MODNAME@ and the name from the user data area is substituted. If some of your members are linked in the same way, you can use this technique to set up a single skeleton link member to handle several different members.

$CCFU009

Provides a sample exit that substitutes the SYSUID for *UID and the RACF PASSWORD for *PSW. You can code the skeleton job card with NOTIFY=*UID and USER=*UID,PASSWORD=*PSW for RACF.

$CCFU015

Provides a sample exit to display a panel requesting a user-specified change request number during Option 1 (OPEN) processing. You must specify leading zeros in the change request number.

JCL and EXEC Skeletons

The JCL skeletons and EXECs reside in the LIBSCCF MACLIB created during installation of LIB/CCF. Skeletons are named $CC*x*Jnnn, where *x* is a letter indicating the operating system, either C for VM/ESA, V for VSE/ESA, or F for z/OS and OS/390.

Review each of the JCL skeletons to determine which of them require modification to conform to your site requirements. Your LIB/CCF site does not reference every skeleton. It is, however, a good practice to check each one so that if you later update your system tables to reference different skeletons, you do not have to go back to see if you must change anything else.

The skeletons contain comments to help you with modification. Possible changes are:

- For VM/ESA, the assembly skeleton $CCCJ010 executes the CCFASM EXEC. The CCFASM EXEC issues a GLOBAL MACLIB DMSSP CMSLIB for VM/ESA; VM/ESA sites must replace that GLOBAL MACLIB statement with GLOBAL MACLIB DMSGPI. Additionally, the CCFASM EXEC executes the H Assembler (HASM). If your site does not have the H Assembler, modify the CCFASM EXEC to execute the F Assembler (ASM).
- For VSE/ESA skeletons, use the format of the LIBDEF that is compatible with your release of VSE/ESA.
- For z/OS and OS/390 skeletons, add a STEPLIB DD statement to the skeletons where appropriate to avoid S806 abends.
- The VSE/ESA skeletons that execute the VSE/ESA batch CA Librarian program use the default name of LIBRPROG. If CA Librarian was renamed at installation time, update the skeletons with the correct program name.
- The z/OS and OS/390 test/production compare ($CCFJ101) executes the z/OS and OS/390 Comparator program COMP2, which can be renamed at installation time. If it was renamed, it is necessary to update skeleton $CCFJ101 with the appropriate program name.
- For VSE/ESA skeletons, change references to MASTER/SYS004 and JOBSTR/SYS008 to the filenames and logical units specified in your LIBGEN macro when the VSE/ESA batch CA Librarian was installed. You must provide extent information for the JOBSTR file. Remove comments on the DLBL statements to avoid editing errors during dynamic substitution.
- If you have LIB/AM installed, you can remove the CA Librarian step and use LIB/AM in the compile/assembly. See the CA Librarian LIB/AM (VSE/ESA) or the CA Librarian LIB/AM (z/OS and OS/390) documentation for details on using LIB/AM.

$CCxJ000 Skeleton

Skeleton JOB card for batch submissions.

$CCFJ000

Used in the VM/ESA, z/OS and OS/390, and VSE/ESA operating systems. Skeleton job card for batch submissions. This skeleton is filled in with the information entered in Option 0.2 (JOBCARD). The symbolics in this skeleton are:

- &$CCFJC1—First line of job card.
- &$CCFJC2—Second line of job card.
- &$CCFJC3—Third line of job card.
- &$CCFJC4—Fourth line of job card.

$CCxJ001 Skeleton

Skeleton JCL for CA Librarian Utility COPY - production to test.

Used in the z/OS and OS/390 and VSE/ESA operating systems. Skeleton JCL for CA Librarian Utility COPY - production to test. Copies all archive levels of a module. The following symbolics are used in this skeleton:

- &PRODMCD—Production master MCD code.
- &TESTMCD—Test master MCD code.
- &TSTATUS—Status specified for the test master.
- &MODULENM—Module name.
- &PMAST—DSN of production master file.
- &TMAST—DSN of test master file.
- &AFOLIBR—CA Librarian batch program name.
- &PVSER—Volume serial number of the disk the production master file is on (used by VSE/ESA).
- &TVSER—Volume serial number of the disk the test master file is on (used by VSE/ESA).

$CCxJ002 Skeleton

Skeleton JCL for CA Librarian Utility COPY - test to production.

Used in the z/OS and OS/390 and VSE/ESA operating systems. Skeleton JCL for CA Librarian Utility COPY - test to production. Copies all archive levels of a module. The following symbolics are used in this skeleton:

- &PRODMCD—Production master MCD code.
- &TESTMCD—Test master MCD code.
- &PSTATUS—Status specified for the production master.
- &MODULENM—Module name.
- &VERS—Current archive version on production.
- &PRODPSWD—Member password on production master.
- &TESTPSWD—Member password on test master.
- &PMAST—DSN of production master file.
- &TMAST—DSN of test master file.
- &AFOLIBR—CA Librarian batch program name.
- &PVOL—Volume serial number of the disk the production master file is on (used by VSE/ESA).
- &TVOL—Volume serial number of the disk the test master file is on (used by VSE/ESA).

$CCxJ003 Skeleton

Skeleton JCL for CA Librarian production application system LKED.

Used in the VM/ESA, z/OS and OS/390, and VSE/ESA operating systems. In VM/ESA, skeleton $CCCJ201 is tailored before this one to provide required variables. It contains skeleton JCL for production application system link requests. The following symbolics are used in this skeleton:

- &OPTS—Link-edit parm options.
- &LPOBJ—Production object library.
- &LPLOAD—Production load library.
- &LPVCORE—Production core image library (used by VSE/ESA).
- &LPOVOBJ—Production object library (used by VSE/ESA).
- &CLACT—Link-edit action card (used by VSE/ESA).
- &$CCCT900—Table containing VSE/ESA linkage control cards.
- &$CCCT901—Table containing VSE/ESA phase names.

$CCxJ004 Skeleton

Skeleton JCL for special CA Librarian copy - test to production.

Used in the z/OS and OS/390 and VSE/ESA operating systems. It contains skeleton JCL for LIBRCOPY (z/OS and OS/390) or LIBDCOPY (VSE/ESA) to copy from test to production. Copies only the current archive level of a module. The following symbolics are used in this skeleton:

- &PRODMCD—Production master MCD code.
- &TESTMCD—Test master MCD code.
- &PSTATUS—Status specified for the production master.
- &MODULENM—Module name.
- &VERS—Current archive version on production master.
- &PRODPSWD—Member password on production master.
- &TESTPSWD—Member password on test master.
- &PMAST—DSN of production master file.
- &TMAST—DSN of test master file.
- &AFOLIBR—CA Librarian batch program name.
- &PVOL—Volume serial number of the disk the production master file is on (used by VSE/ESA).
- &TVOL—Volume serial number of the disk the test master file is on (used by VSE/ESA).

$CCxJ005 Skeleton

Skeleton JCL for COPY of object - test to production.

Used in the VM/ESA, z/OS and OS/390, and VSE/ESA operating systems. In VM/ESA, skeleton $CCCJ200 is tailored before this one to provide required variables. It contains skeleton JCL for copy of object - test to production. The following symbolics are used in this skeleton:

- &TOBJECT—Test object library.
- &POBJECT—Production object library.
- &MODULENM—Module name.
- &VPOBJECT—VSE/ESA production object library.
- &VTOBJECT—VSE/ESA test object library.

$CCxJ006 Skeleton

Skeleton JCL for special CA Librarian copy - production to test.

Used in the z/OS and OS/390 and VSE/ESA operating systems. It contains skeleton JCL for LIBRCOPY (z/OS and OS/390) or LIBDCOPY (VSE/ESA) - production to test. Copies only the current archive level of a module. The following symbolics are used in this skeleto EXEC skeletonn:

- &PRODMCD—Production master MCD code.
- &TESTMCD—Test master MCD code.
- &PSTATUS—Status specified for the production master.
- &MODULENM—Module name.
- &PMAST—DSN of production master file.
- &TMAST—DSN of test master file.
- &AFOLIBR—CA Librarian batch program name.
- &PVSER—Volume serial number of the disk the production master file is on (used by VSE/ESA).
- &TVSER—Volume serial number of the disk the test master file is on (used by VSE/ESA).

$CCxJ007 Skeleton

Skeleton JCL for production to backup load COPY.

Used in the CMS, MVS, and VSE operating systems. In CMS, skeleton $CCCJ200 is tailored before this one to provide required variables. It contains skeleton JCL to backup production executable module. The following symbolics are used in this skeleton:

- &BKLOAD—Production backup load library.
- &LPLOAD—Production load library.
- &LPLVCORE—Production core image library.
- &LBLVCORE—Backup core image library.

$CCxJ009 Skeleton

Skeleton used to write SYSIN data to JCL stream.

Used in the VM/ESA, z/OS and OS/390, and VSE/ESA operating systems. It is the skeleton used for writing SYSIN data to JCL. The following symbolic is used in this skeleton:

- &JCLREC—SYSIN record.

$CCxJ010 Skeleton

Skeleton JCL for production source assembly.

Used in the VM/ESA, z/OS and OS/390, and VSE/ESA operating systems. In VM/ESA, skeleton $CCCJ200 is tailored before this one to provide required variables. It contains skeleton JCL for production source assembly. The following symbolics are used in this skeleton:

- &MODULENM—Module name.
- &PRODPSWD—Password on production master file.
- &PRODMCD—Production master file MCD code.
- &POBJECT—Production object library.
- &PMAST—Production master file name.
- &AFOLIBR—CA Librarian batch program name.
- &PVOL—Volume serial number of the disk the production master file is on (used by VSE/ESA).

$CCxJ011 Skeleton

Skeleton JCL for production source COBOL compile.

Used in the VM/ESA, z/OS and OS/390, and VSE/ESA operating systems. In VM/ESA, skeleton $CCCJ200 is tailored before this one to provide required variables. It contains skeleton JCL for production source COBOL compile. The following symbolics are used in this skeleton:

- &MODULENM—Module name.
- &PRODPSWD—Password on production master file.
- &PRODMCD—Production master file MCD code.
- &POBJECT—Production object library.
- &PMAST—Production master file name.
- &AFOLIBR—CA Librarian batch program name.
- &PVOL—Volume serial number of the disk the production master file is on (used by VSE/ESA).

$CCxJ012 Skeleton

Skeleton JCL for production source PL1 compile.

Used in the VM/ESA, z/OS and OS/390, and VSE/ESA operating systems. In VM/ESA, skeleton $CCCJ200 is tailored before this one to provide required variables. It contains skeleton JCL for production source PL1 compile.

The following symbolics are used in this skeleton:

- &MODULENM—Module name.
- &PRODPSWD—Password on production master file.
- &PRODMCD—Production master file MCD code.
- &POBJECT—Production object library.
- &PMAST—Production master file name.
- &AFOLIBR—CA Librarian batch program name.
- &PVOL—Volume serial number of the disk the production master file is on (used by VSE/ESA).

$CCxJT10 Skeleton

Skeleton JCL for test source assembly.

$CCxJT11 Skeleton

Skeleton JCL for test source COBOL compile.

$CCxJT12 Skeleton

Skeleton JCL for test source PL1 compile.

$CCFJ030 Skeleton

Skeleton JCL for DB2 for z/OS and OS/390 precompile and assembly.

Used in the z/OS and OS/390 operating system. It contains skeleton JCL for DB2 for z/OS and OS/390 precompile and assembly. The following symbolics are used in this skeleton:

- &AFOLIBR—CA Librarian batch program name.
- &MODULENM—Member name.
- &$CCFDB2—DB2 for z/OS and OS/390 subsystem name.
- &ITT—Target library type.
- &ITMCD—Target master file MCD.
- &IHMCD—History master file MCD.
- &ITMAST—Target master file.
- &IHMAST—History master file.
- &IMTO—z/OS and OS/390 target object library.
- &IMTD—z/OS and OS/390 target DBRM library.
- &WO—Change request number.
- &CCFID—CCFID.

$CCFJ031 Skeleton

Skeleton JCL for DB2 for z/OS and OS/390 precompile and COBOL compile.

Used in the z/OS and OS/390 operating system. It contains skeleton JCL for DB2 for z/OS and OS/390 precompile and COBOL compile. The following symbolics are used in this skeleton:

- &AFOLIBR—CA Librarian batch program name.
- &MODULENM—Member name.
- &$CCFDB2—DB2 for z/OS and OS/390 subsystem name.
- &ITT—Target library type.
- &ITMCD—Target master file MCD.
- &IHMCD—History master file MCD.
- &ITMAST—Target master file.
- &IHMAST—History master file.
- &IMTO—z/OS and OS/390 target object library.
- &IMTD—z/OS and OS/390 target DBRM library.
- &WO—Change request number.
- &CCFID—CCFID.

$CCFJ032 Skeleton

Skeleton JCL for DB2 for z/OS and OS/390 precompile and PL1 compile.

Used in the z/OS and OS/390 operating system. It contains skeleton JCL for DB2 for z/OS and OS/390 precompile and PL1 compile. The following symbolics are used in this skeleton:

- &AFOLIBR—Batch CA Librarian name.
- &MODULENM—Member name.
- &$CCFDB2—DB2 for z/OS and OS/390 subsystem name.
- &ITT—Target library type.
- &ITMCD—Target master file MCD.
- &IHMCD—History master file MCD.
- &ITMAST—Target master file.
- &IHMAST—History master file.
- &IMTO—z/OS and OS/390 target object library.
- &IMTD—z/OS and OS/390 target DBRM library.
- &WO—Change request number.
- &CCFID—CCFID.

$CCxJ100 Skeleton

Skeleton JCL step for automatic job checkoff.

Used in the VM/ESA, VSE/ESA, and z/OS and OS/390 operating systems. It contains skeleton JCL for automatic job checkoff. This skeleton has no symbolics.

$CCxJ101 Skeleton

Skeleton JCL/EXEC for test to production compare.

Used in the VM/ESA, z/OS and OS/390, and VSE/ESA operating systems. It contains skeleton JCL/EXEC for test/production compare. The z/OS and OS/390 skeleton executes Comparator, which is site-named at installation time. Update this skeleton with the appropriate program name. The following symbolics are used in this skeleton:

- &MODULENM—Module name.
- &PRODPSWD—Password on production master file.
- &PRODMCD—Production master file MCD code.
- &POBJECT—Production object library.
- &PMAST—Production master file name.
- &AFOLIBR—CA Librarian batch program name.
- &PVOL—Volume serial number of the disk the production master file is on (used by VSE/ESA).
- &TMAST—Test master file.
- &TESTMCD—Test master file MCD.
- &TVOL—Volume serial number of the disk the test master file is on (used by VSE/ESA).
- &LOLEVEL—Original level of module.

$CCFJ103 Skeleton

Skeleton JCL for DB2 for z/OS and OS/390 bind.

Used in the z/OS and OS/390 operating system. It contains skeleton JCL for production DB2 for z/OS and OS/390 bind. The following symbolics are used in this skeleton:

- &IMTD—Target DBRM library.
- &$CCFDB2—DB2 for z/OS and OS/390 subsystem name.

$CCCJ200 Skeleton

Establish VM/ESA global variables for use in later job steps.

Used in the VM/ESA operating system. It establishes VM/ESA global variables for use in later job steps. The following symbolics are used in this skeleton:

- &PMAST—Production master file name.
- &PLKADR—CP link address for production master file disk.
- &PLKID—Owner of the production master file disk.
- &PPASS—Multiwrite password for the production master file disk.
- &PRODMCD—MCD code for the production master file.
- &SMAST—CCF system master file name.
- &SLKADR—CP link address for CCF system master file disk.
- &SLKID—Owner of the CCF system master file disk.
- &SPASS—Multiwrite password for the CCF system master file disk.
- &SYSMCD—MCD code for the CCF system master file.
- &POWNER—Owner of the production object disk.
- &POADDR—CP link address of the production object disk.
- &POMODE—CP link mode of the production object disk.
- &POPASS—Password to link &POMODE to the production object disk.
- &POTXT—Name of a production TXTLIB.
- &TOWNER—Owner of the test object disk.
- &TOADDR—CP link address of the test object disk.
- &TOMODE—CP link mode of the test object disk.
- &TOPASS—Password to link &POMODE to the test object disk.
- &TOTXT—Name of a test TXTLIB.
- &MODULNM—Name of the module to compile.
- &PRODPSWD—Password of the module to compile.

$CCCJ201 Skeleton

Establish VM/ESA global variables for use in later job steps.

Used in the VM/ESA operating system. It establishes VM/ESA global variables for use in later job steps. The following symbolics are used in this skeleton:

- &SMAST—CCF system master file name.
- &SLKADR—CP link address for CCF system master file disk.
- &SLKID—Owner of the CCF system master file disk.
- &SPASS—Multiwrite password for the CCF system master file disk.
- &SYSMCD—MCD code for the CCF system master file.
- &LPLOWNER—Owner of the production load disk.
- &LPLPASS—Password to link &LPLMODE to the production load disk.
- &LPLADDR—CP link address of the production load disk.
- &LPLMODE—CP link mode of the production load disk.
- &LPLLOAD—Load library name for production load library.
- &LPOOWNER—Owner of the production object disk.
- &LPOPASS—Password to link &LPLMODE to the production object disk.
- &LPOADDR—CP link address of the production object disk.
- &LPOMODE—CP link mode of the production object disk.
- &LPOTXT—TXTLIB name for a production object library.
- &LBLOWNER—Owner of the backup production load disk.
- &LBLPASS—Password to link &LPLMODE to the backup production load disk.
- &LBLADDR—CP link address of the backup production load disk.
- &LBLMODE—CP link mode of the backup production load disk.
- &LBLLOAD—Load library name for backup production load library.
- &CLLET—LET option for LKED.
- &CLNCAL—NCAL option for LKED.
- &CLRENT—RENT option for LKED.
- &CLREUS—REUS option for LKED.
- &CLREFR—REFR option for LKED.
- &CLCLEAR—CLEAR/NOCLEAR option for LOAD.
- &CLAUTO—AUTO/NOAUTO option for LOAD.
- &CLLIBE—LIBE/NOLIBE option for LOAD.

- &CLRLDS—RLDSAVE option for LOAD.
- &CLORIG—Origin value for LOAD.
- &CLMAP—MAP/NOMAP option for GENMOD.
- &CLSTR—STR/NOSTR option for GENMOD.
- &CLSYS—OS/DOS/ALL option for GENMOD.
- &CLSYSX—SYSTEM option for GENMOD.

$CCFJ200 Skeleton

Skeleton JCL for CA MetaCOBOL+.

Used in the z/OS and OS/390 operating system. It contains skeleton JCL for Advantage CA-MetaCOBOL+. The following symbolics are used in this skeleton:

- &MODULENM—Module name.
- &PRODPSWD—Password for production module.
- &PRODMCD—Production master file MCD code.
- &POBJECT—Production object library.
- &PMAST—Production master file name.
- &AFOLIBR—CA Librarian batch program name.

$CCFJ201 Skeleton

Skeleton JCL for CA APCDOC.

Used in the z/OS and OS/390 operating system. It contains skeleton JCL for Unicenter CA-APCDOC for JCL procs. The following symbolics are used in this skeleton:

- &MODULENM—Module name.
- &PRODPSWD—Password for production module.
- &PRODMCD—Production master file MCD code.
- &POBJECT—Production object library.
- &PMAST—Production master file name.
- &AFOLIBR—CA Librarian batch program name.

Link-Edit Considerations

Two Option 12.0 (PROFILE) keywords affect the way link-edits are performed in LIB/CCF. They are LKED and LNKNTEGRTY. See the section titled System Profile Table in Chapter 3, “Administrator Functions in LIB/CCF,” for details on each keyword.

From Option 6 (LINK), the programmer is prompted for the CCF Change Request ID, the target operating system, and the function to perform (a link or DB2 for z/OS and OS/390 bind). For details on the DB2 for z/OS and OS/390 bind process, see the section titled Initiating a Production Bind Request in the *LIB/CCF User Guide*. The panel defaults to L for link. The programmer sees a list of production master files and their associated production executable libraries. He must then enter the number of the pair that defines the master file containing the associated linkage editor input and the executable library where the modules should be linked.

The programmer must also enter either a LKED SYSIN MEMBER, a LKED JCL MEMBER, an OVERRIDE SKELETON, or a MODULE NAME. Information about each field follows:

LKED SYSIN MEMBER

This field is:

- Mutually exclusive with LKED JCL MEMBER and MODULE NAME.
- Must reside on the production master file selected in Option 6.
- Specifying this field displays the Linkage Editor Options Selection panel. To choose an option, place an **X** next to it on the display.

For z/OS and OS/390 jobs:

- Must contain the required link-edit INCLUDE statements that reference a ddname of OBJECT and a link-edit NAME statement specifying the REPLACE(R) option. Other link-edit control statements are optional as required.
- This member is used as SYSLIN to the default link-edit skeleton ($CCFJ003) or, if specified, the override link-edit skeleton.
- Selected link-edit options are inserted into the PARM= field of the EXEC statement in the JCL stream that is submitted.
- Automatic load module backup (if specified in LCDF) is based on the names on the link-edit NAME statements.

For VSE/ESA jobs:

- Must contain the required link-edit PHASE and INCLUDE statements. Other link-edit control statements are optional.
- This member is used as input to the default link-edit skeleton ($CCVJ003) or, if specified, the override link-edit skeleton.
- Selected link-edit options are included on the generated link-edit ACTION control statement.

- Automatic phase backup (if specified) is based on the names on the PHASE statements.

For VM/ESA jobs:

- Not applicable for the default link-edit skeleton ($CCCJ003) that performs a LOAD and GENMOD for the MODULE NAME supplied.

LKED JCL MEMBER

This field is:

- Mutually exclusive with LKED SYSIN member and override skeleton.
- Must reside on production master file selected.
- Must be a complete link-edit JCL stream.
- LIB/CCF submits as is for execution.
- Job name is derived from the job card included in this member.
- Automatic load module backup is not performed unless a step is inserted into the JCL stream to accomplish it.

OVERRIDE SKELETON

This field is:

- Mutually exclusive with LKED JCL MEMBER.
- Must reside in the ISPF skeleton library allocated to ISPSLIB.
- Requires that a LKED SYSIN MEMBER or MODULE NAME also be supplied.
- Specifies a skeleton used in place of the default link-edit skeleton.
- Automatic executable module backup (if specified in LCDF) is based on the NAME statement (for z/OS and OS/390), the PHASE statement (for VSE/ESA), or the MODULE NAME or names specified in the LKED SYSIN MEMBER (for VM/ESA).

Module Name

This field is:

- Mutually exclusive with SYSIN MEMBER and LKED JCL MEMBER.
- The module name supplied replaces the &CCFMOD variable that appears in the default VM/ESA link-edit skeleton ($CCCJ003). You can also code this variable in an z/OS and OS/390 or VSE/ESA skeleton. However, as distributed, the z/OS and OS/390 and VSE/ESA skeletons do not use it.
- Specifying this field displays the Linkage Editor Options Selection panel. To choose an option, place an **X** next to it on the display. Selected options are placed in the PARM= field for z/OS and OS/390, the ACTION statement for VSE/ESA, and inserted into the LOAD or GENMOD command for VM/ESA.
- Automatic executable module backup (if specified in LCDF) is based on the module name supplied here.

Customizing Panels and Reports

LIB/CCF is distributed with an CA Earl interface. Advantage CA-Earl is a report generation system that uses a high-level programming language to produce customized reports. The LIB/CCF-Advantage CA-Earl interface provides access to information LIB/CCF collected and stored on the System Master file. See the “LIB/CCF-Advantage CA-Earl Interface” chapter for details.

In addition to the LIB/CCF-Advantage CA-Earl interface, Computer Associates provides a macro to allow modification to LIB/CCF panel and report titles. After assembling and link editing the macro with the overrides, both the panel titles and the report headings are affected. It is not necessary to modify the actual LIB/CCF panels. Reports printed online from LIB/CCF reflect titles changed by the macro. The following table describes the panel titles that the macro can modify (other titles on the panels are not modifiable).

Option	Panel	Title Field	Macro Label	Max Length
1-OPEN	$CCFPN01	REQUESTOR	TITLE1	11
2-ASSIGN	$CCFPN02	PHONE	TITLE2	11
3-DISPLAY	$CCFPN03	MANAGER	TITLE3	11
		DEPARTMENT	TITLE4	11
		COORDINATOR	TITLE5	11
		PHONE	TITLE6	11
2-ASSIGN 3-DISPLAY	$CCFPN02 $CCFPN03	MGR COMMENT	TITLE7	11
7-CLOSE	$CCFPN07	ANALYSIS	TITLE8	8
7-CLOSE	$CCFPN19	ROUTINES	TITLE9	11
		AFFECTED	TITLE10	11
7-CLOSE	$CCFPN20	SYSTEMS TESTS MODIFIED ON	TITLE11	25
		SYSTEMS TESTS VERIFIED ON	TITLE12	25
		SYSTEMS TESTS ADDED OR MOD	TITLE13	26
		NEW USER INPUT	TITLE14	14
		NEW USER OUTPUT	TITLE15	15
		DOCUMENTATION CHANGES	TITLE16	21
4-LOGOUT	$CCFPN14	SYSTEM	TITLE17	11
5-LOGIN	$CCFPN15	DESCRIPTION	TITLE18	11
		ABSTRACT	TITLE19	11
5-LOGIN	$CCFPN15	DESCRIPTION	TITLE20	11
		OF CHANGES	TITLE21	11

The CCFTGEN EXEC assembles and link edits the $CCFCOMT macro and stores the resulting load module in the LIBLCCF LOADLIB. The CCFTGEN EXEC obtains the assembler input source from $CCFCOMT ASSEMBLE. Both CCFTGEN EXEC and $CCFCOMT ASSEMBLE are unloaded to the installer's A-disk at LIB/CMS installation time. To change a title, you must first edit the $CCFCOMT ASSEMBLE file with the title. Then execute CCFTGEN EXEC that performs the assembly and link.

Note: Remember the following guidelines when editing $CCFCOMT ASSEMBLE:

- The title appears on the panels and in the reports *exactly* as entered in the override. It can be right or left justified, contain embedded blanks, and be lower or upper case.
- You must delimit titles with apostrophes.

Specify the macro name $CCFCOMT on or before column 16. To continue parameters, you must code any non-blank in column 72 and continue the next parameter starting in column 16 of the next line.

Additionally, you can modify the following panels as indicated below:

$CCFPN00

Primary LIB/CCF panel. This is the panel where you select the LIB/CCF functions. You can modify this panel to remove options that your site does not use.

HELP panel: $CCFH005

$CCFPN08

This is the Option 8 panel to select a report. You can modify this panel to remove reports or add additional reports.

HELP panel: $CCFH080

$CCFPN18

This is the Option 6 panel for the LKED options specification. You can modify this panel to specify different default options.

HELP panel: None

Execution

Select Option **C** on the CA Librarian selection menu (Option L from the main ISPF menu) to execute LIB/CCF.

PF Keys and the Jump Command

You can display and modify PF key definitions for LIB/CCF by using the ISPF KEYS command from any LIB/CCF panel or Option 0.1. The PF key settings affect the current and subsequent LIB/CCF sessions for the user. However, changing the PF key settings while in LIB/CCF does *not* affect the PF key settings for ISPF. LIB/CCF maintains its own PF key settings in the user's ISPF profile. Conversely, changing PF key settings while in ISPF does not affect LIB/CCF PF key settings.

The default settings are:

- PF1 and PF13 (- HELP)
- PF2 and PF14 (- SPLIT)
- PF3 and PF15 (- END)
- PF4 and PF16 (- RETURN)
- PF5 and PF17 (- RFIND)
- PF6 and PF18 (- RCHANGE)
- PF7 and PF19 (- UP)
- PF8 and PF20 (- DOWN)
- PF9 and PF21 (- SWAP)
- PF1and and (PF22 - LEFT)
- PF11 and PF23 (- RIGHT)
- PF12 and PF24 (- CURSOR)

LIB/CCF supports a jump command that enables a direct jump to a specified option. The syntax is:

`=`*`option`*`[.`*`option`*`]`

Examples: =3 or =8.5

Job Card Specification

The JCL job statement skeleton is used when LIB/CCF jobs are submitted for batch execution. Appropriately modify the job statement to reflect the correct accounting information, programmer name, job and message class, and so on.

LIB/CCF-ISPF(VM/ESA) supports one set of job information for each of the three supported operating systems. Option 0.2 (VM/ESA), 0.3 (z/OS and OS/390), and 0.4 (VSE/ESA) contain the same information with the exception of 0.3 (z/OS and OS/390), which contains the DB2 for z/OS and OS/390 subsystem name. The fields are as follows:

JOB STATEMENT

These statements are passed to the batch machine as a job card. There can be up to five cards. They must begin on the first line.

SPOOL MACHINE ID

Specify the name of the machine that the job is spooled to. This field is required.

SPOOL TAG

Specify information to TAG the punched job. This field is optional. However, your site can require tag information.

PROMPT AT SUBMIT

Specify YES or NO.

YES

Displays the Job Card Specification panel before each submit, providing an opportunity to modify information for that job.

If a user submits their own job, he can modify any information.

If a user is requesting that the control group submit the job (that is, a batch login), the user can modify only the operating system (unless the user specified a DEFAULT SYSTEM).

If a control group member is submitting the job as per a user's request, you can modify any information except the operating system.

NO

Suppresses the display of the Job Card Specification panel. The job uses the default job information.

DEFAULT SYSTEM

Leave this field blank to display the Machine Type Specification panel before each submit. Alternatively, specify VM/ESA, z/OS and OS/390, or VSE/ESA to suppress the display of the Machine Type Specification panel and to establish your default target system.

DB2 SUBSYSTEM

Option 0.3 (z/OS and OS/390) only. Specify the DB2 for z/OS and OS/390 subsystem to use for DB2 for z/OS and OS/390 binds. The default is DSN. This field is required only if LIB/CCF supports DB2 for z/OS and OS/390 at your site.

For z/OS and OS/390 sites using RACF, hard code the RACF user and password on the jobcard skeleton using Option 0.3.

System Error Processing

LIB/CCF-ISPF(VM/ESA) errors fall generally into three categories:

- The first is an ISPF dialog error. When this happens, contact your LIB/CCF administrator immediately. PF1 displays more information.
- The second occurs when LIB/CCF executes one of the LIBCMS commands like LIBIMP or LIBEXP. First you get a brief message from the command itself, for example, Unsuccessful LIBEXP. Then, when you get back to the LIB/CCF panel, you see a message like Update error. Press PF1 to display a longer message that identifies the VM/ESA file that contains the detailed error messages.

The third error is a file allocation error. In addition to the various LIB/CCF master files, LIB/CCF uses a number of temporary data sets for work files, which it allocates and deletes as required. Occasionally, an error in LIB/CCF causes subsequent LIB/CCF executions to fail because a work file fails to get deleted. You can get more information about the data set that is failing allocation by pressing PF1. Exiting LIB/CCF then reentering LIB/CCF usually clears the problem. In some cases, you might have to explicitly issue a delete of the file in question.

Chapter 15: Customizing Your System: LIB/CCF CA Vollie

You can tailor LIB/CCF-AllFusion CA-Vollie to your site's needs in a number of ways over and above the options available to the system administrator through the LIB/CCF Table Processors. A number of exit points and sample exits are available. You must modify JCL skeletons to fit your site's requirements. You can also change panels and reports.

This section contains the following topics:

User Exits (see page 297)
JCL Skeletons (see page 307)
Link-Edit Considerations (see page 315)
Customizing Panels (see page 316)
Customizing Reports (see page 318)
Execution (see page 319)
PF Keys and Commands (see page 319)
Job Card Specification (see page 320)
System Error Processing (see page 320)

User Exits

LIB/CCF executes the user exits if they are present in the VOICE library where CCF was installed. The System Services File provides sample exit programs. In addition to the variables that LIB/CCF passes to the exits, global variables @G55 through @G59 inclusive, are available for exit processing. Following are the specifications for the exit procs:

$CCVU000

Invoked by service program $CCVS001. It can return up to 42 bytes of user data in the @1 variable that are placed on the module tracking record and any module movement records or job submission records that the logout request can create.

$CCVU001

Invoked by function program $CCVP006. It can return up to 56 bytes of user data in the @1 variable that is placed on the system link record and any job submission records that the LINK request can create.

$CCVU002

Invoked by function program $CCVP005. It can return up to 42 bytes of user data in the @1 variable that is placed on the module movement record and job submission records that the login request can create. If this exit is not used and exit $CCVU000 supplied data during the logout request, then the existing user data is carried forward to the module movement record and job submission record. The exit has access to the following information:

- @2—Member name
- @3—The master file in which the member currently resides.

$CCVU003

Invoked by function program $CCVP004 after the data is entered and verified on the logout panel. This exit has access to the following information:

- @1—Member name.
- @2—Production master information:
  - 44 character data set name
  - 7 character FCT name
  - 1 character file organization indicator (B for BDAM, V for VSAM)
  - 6 character volume serial number
  - 7 character VSAM catalog name (blank if BDAM)
  - 4 character MCD base code
  - 5 character security status
  - 1 character member copy indicator (S for LIBRCOPY, U for Utility Copy)
  - 5 character LOCK status.
- @3—Test master information (same as @2 information, listed above).
- @4—Message area.

The exit can place an error message in the message variable @4 and redisplay the Logout panel with the message by exiting with a non-zero return code. For example:

```
@4 = 'MODULE NAME DOES NOT CONFORM TO STANDARDS'
@EXIT 4
```

$CCVU004

Invoked by service programs $CCVS007 and $CCVS008 before the history create and history update panels display, creating an opportunity to provide data for this panel. The exit can standardize information used for history create and update. The MODIFY function named below pertains to the modification of existing history information through the Option 5 HISTORY command. A return code of 4 from the exit suppresses the display of the history create and update panel.

This exit can modify the data fields as follows:

- CREATE or MODIFY Function:
  - @1—System
  - @2—Description
  - @3—Language
  - @4—Line 1 of abstract
  - @10—'C' (for CREATE) or 'M' (for MODIFY)
- UPDATE Function:
  - @1—Member name
  - @2—Production master file ID
  - @3—CR ID
  - @4—Line 1 of description
  - @10—'U' (for UPDATE)

$CCVU005

Invoked by service programs $CCVS007 and $CCVS008 after the history create and history update panels are filled in and verified. The MODIFY function named below pertains to the modification of existing history information through the Option 5 HISTORY command. A return code of 4 from the exit suppresses the display of the history create and update panel. This exit has access to the data fields as follows:

- CREATE Function:
  - @1—System
  - @2—Description
  - @3—Language
  - @4—Line 1 of abstract
  - @5—Message area
  - @10—'C' (for CREATE)
- MODIFY or UPDATE Function:
  - @1—Member name
  - @2—Production master file-id
  - @3—CR ID
  - @4—Line 1 of description
  - @5—Message area
  - @10—'M' (for MODIFY), 'U' (for UPDATE)

The exit can place an error message in the message variable @5 and redisplay the history create or history update panel with the message by exiting with a return code corresponding to the variable in error. For example, for the CREATE function:

```
@5 = 'DESCRIPTION DOES NOT CONFORM TO STANDARDS'
@EXIT 2
```

places the cursor in the @2 (that is, description) field.

$CCVU006

Invoked by function program $CCVP005 after the login function (Option 5) completes successfully. The exit can provide the opportunity for special reports and messages. The following variables are provided to the exit:

- @1—Module name
- @2—Production Master File-id
- @3—Change Request ID
- @4—Test Master File-id
- @5—Command as entered on the panel

The exit can place an error message in the message variable @6 and redisplay the login panel with the message by exiting with a return code of 4.

$CCVU007

Invoked by service program $CCVS005 before the update to lock the test member. Member CCFLMI contains the update stream to lock the member (-MCD and -SEL card). The exit can change or add to this stream. This is also the point where the member can automatically be EXPORTed from the CA Vollie library to the test library if you are using the CA Vollie library for development. The following variables are provided to the exit:

- @1—Test Master File information:
  - 44 character data set name
  - 7 character FCT name
  - 1 character file organization indicator (B for BDAM, V for VSAM)
  - 6 character volume serial number
  - 7 character VSAM catalog name (blank if BDAM)
  - 4 character MCD base code
  - 5 character security status
  - 1 character member copy indicator (S for LIBRCOPY, U for Utility Copy)
  - 5 character LOCK status
- @2—Member name
- @3—Member password
- @4—Master file MCD

$CCVU008

Invoked by service program $CCVS006 before processing the link-edit INPUT member or link-edit JCL member. The exit can verify or modify data in the member.

- @1—User data area from SLR

$CCVU010

Invoked by function program $CCVP009 when the PROCESS command is entered and the movement of the member is to production. The exit can restrict the PROCESS command to a subset of the control group when movement is to production.

If the PROCESS command is rejected, the exit must return an error message in the @7 variable and exit with a return code of 4. If the exit returns with a return code of 0, CCF processes the command after returning from the exit. The following variables are provided to the exit:

- @1—Member name
- @2—Production Master File name
- @3—Test Master File information
  - 44 character data set name
  - 7 character FCT name
  - 1 character file organization indicator (B for BDAM, V for VSAM)
  - 6 character volume serial number
  - 7 character VSAM catalog name (blank if BDAM)
  - 4 character MCD base code
  - 5 character security status
  - 1 character member copy indicator (S for LIBRCOPY, U for utility Copy)
  - 5 character LOCK status
- @4—Change Request ID
- @5—CCF ID
- @6—VERS/user data
- @7—Message variable

$CCVU011

Invoked by function program $CCVP004 after processing the logout request. The exit can allow the member to be IMPORTed into the AllFusion CA-Vollie library from the test library automatically. The variables:

- @1—Member name
- @2—Test master file information:
  - 44 character data set name
  - 5 character security status
  - 1 character member COPY indicator (S for LIBRCOPY, U for utility Copy)
  - 5 character LOCK status
- @3—Change Request ID
- @4—Master file MCD

$CCVU015

Invoked by function $CCVP001 when a change request is opened using option 1 (OPEN). This exit can assign site-specified change request identifiers. The variables are:

- @1—The text of an error message. If this parameter is not null, the request identifier returned by a previous call to this exit was invalid. The exit can display the error message and give the user another opportunity to enter a valid request identifier.
- @2—Request ID (IN/OUT)
- @3
  - Position 1 (20 characters): Requestor.
  - Position 21 (20 characters): Requestor's phone number .
  - Position 41 (20 characters): Requestor's manager.
- @4
  - Position 1 (20 characters): Application.
  - Position 21 (20 characters): Department.
  - Position 41 (20 characters): Co-ordinator.
  - Position 21 (20 characters): Co-ordinator's phone number .
- @5
  - Position 1: Non-blank if Problem.
  - Position 2: Non-blank if Modification.
  - Position 3: (61 characters) Description line 1.
- @6
  - Position 1 (61 characters): Description line 2.
  - Position 62 (61 characters): Description line 3.
  - Position 123 (61 characters): Description line 4.
- @7
  - Position 1 (61 characters): Description line 5.
  - Position 62 (61 characters): Description line 6.
  - Position 123 (61 characters): Description line 7.
- @8
  - Position 1 (61 characters): Description line 8.
  - Position 62 (61 characters): Description line 9.
  - Position 123 (61 characters): Description line 10.

Sample User Exits

The System Services File provides the following sample exits:

$CCVU000

These exits are similar in that they can provide user data that is placed in the

$CCVU001

LIB/CCF system records. They provide sample exits that display a panel asking for

$CCVU002

the date and time the request should be completed and put this data in the user area. $CCVU001 also asks for the member name to link. Sample exit @CCVU008 uses this information

$CCVU003

Provides a sample exit that prevents certain master files from being selected for logout, based on the CA Vollie SAFE specifications. You can modify this exit to check the CCFID and permit only certain programmers to use certain master files or certain member names.

$CCVU004

Provides a sample exit that fills in the history CREATE panel with default information. To illustrate how panel data can be passed to the next exit, the information put into the description field is saved in the global variable @G40.

$CCVU005

Provides a sample exit that retrieves the data put into the description field by $CCVU004 (stored in the global variable @G40), then compares it to the description field currently on the panel. If the user changed the description, it is changed back. An error message is issued, and a return code is issued to redisplay the panel.

$CCVU007

Provides a sample exit that inserts a -PGMR card into the update stream, automatically putting the LIB/CCF ID on the member as the programmer name when the member is locked.

$CCVU008

Provides a sample exit that uses the name of the member being linked if it was supplied and placed in the user data area by $CCVU001 to replace all occurrences of the symbolic @MODNAME@ in the link-edit INPUT member or JCL member. If several members are linked in the same way, this could set up a skeleton link member that can be used for more than one member by substituting the appropriate member name.

$CCVU010

Provides a sample exit that restricts the PROCESS command to the CCF administrator for the final move to production for use with Q/A libraries. You must modify the sample exit used by your site. Instructions are included as comments. It lets members of the control group process movement requests through the Q/A chain, but permits only the CCF administrator to process the movement request to production.

$CCVU011

Provides a sample exit that imports and saves the test member into the CA Vollie library.

$CCFU015

Provides a sample exit to display a panel requesting a user-specified change request number during Option 1 (OPEN) processing. You must specify leading zeros in the change request number.

JCL Skeletons

The JCL skeletons distributed with LIB/CCF-CA-Vollie reside in the CA Vollie library where LIB/CCF is installed.

Review each of the JCL skeletons to determine which of them require modification to conform to your site requirements. Your LIB/CCF site does not reference every skeleton. It is, however, good practice to check each one so that if you later update your system tables to reference different skeletons, you do not have to go back to see if anything else must be changed.

The skeletons contain comments to help you with modification.

Possible changes:

- Change references to MASTER/SYS004 and JOBSTR/SYS008 to the filenames and logical units specified in your LIBGEN macro when batch CA Librarian was installed. You must provide extent information for JOBSTR files. Remove comments on the JCL statements to avoid editing errors during dynamic substitution.
- Use the format of the LIBDEF that is compatible with your release of VSE/ESA.
- If you have LIB/AM installed, you can remove the CA Librarian step and use LIB/AM in the compile/assembly. See the User Guide for details on using LIB/AM.
- If you have only VSAM master files, you can remove the EXTENT and ASSGN statements for your masters, leaving only the DLBL. However, if you have mixed file types, leave the EXTENT and ASSGN information in the skeletons.

Note: The Master File Definition Table (Option 12.4) contains the field COPY. When you specify U, CA Librarian Utility COPY moves all archive levels of the member. If you specify S, LIBDCOPY moves only the current archive level of the member.

$CCVJ000 Skeleton

Skeleton JOB card for batch submissions.

Skeleton JOB card and LST card for batch submissions. The symbolics for this skeleton are:

- JNAME—JOB name.
- PNAME—Programmer (operator) name.

LIBR--$CCVJ001 Skeleton

Skeleton JCL for CA Librarian Utility COPY - production to test. The symbolics for this skeleton are:

- TAG1—Member name.
- TAG2—Production master file.
- TAG3—Test master file.
- TAG4—Test master MCD.
- TAG5—Production master MCD.
- TAG6—Test master file MCD security status.
- TAG7—Production master volser.
- TAG8—Test master volser.

$CCVJ002 Skeleton

Skeleton JCL for CA Librarian Utility COPY - test to production. The symbolics for this skeleton:

- TAG0—CA Librarian program name.
- TAG1—Member name.
- TAG2—Production master file.
- TAG3—Test master file.
- TAG4—Production master MCD.
- TAG5—Test master MCD.
- TAG6—Production master file MCD security status.
- TAG7—Production master file volser.
- TAG8—Test master file volser.

$CCVJ003 Skeleton

Skeleton JCL for CA Librarian Production application system link-edit. The symbolics for this skeleton are:

- TAG5—Production core image library.
- TAG7—Production relocatable library.

- TAG8—Production backup core image library.
- TAG10—Link-edit options.

$CCVJ004 Skeleton

Skeleton JCL for LIBDCOPY - test to production. The symbolics for this skeleton are:

- TAG0—CA Librarian program name.
- TAG1—Member name.
- TAG2—Production master file.
- TAG3—Test master file.
- TAG4—Production master MCD.
- TAG5—Test master MCD.
- TAG6—Production master file MCD security status.
- TAG7—Production master file volser.
- TAG8—Test master file volser.
- TAG12—Test module password.

$CCVJ005 Skeleton

Skeleton JCL for copy of relocatable module - test to production. The symbolics for this skeleton are:

- TAG1—Member name.
- TAG15—Production relocatable library.
- TAG16—Test relocatable library.

$CCVJ006 Skeleton

Skeleton JCL for LIBDCOPY - production to test. The symbolics for this skeleton are:

- TAG0—CA Librarian program name.
- TAG1—Member name.
- TAG2—Production master file.
- TAG3—Test master file.
- TAG4—Production master MCD.
- TAG5—Test master MCD.

- TAG6—Production master file MCD security status.
- TAG7—Production master file volser.
- TAG8—Test master file volser.
- TAG12—Production module password.

$CCVJ007 Skeleton

Skeleton JCL to back up production phase. The symbolics for this skeleton are:

- TAG1—Module name.
- TAG5—Production core image library.
- TAG8—Backup core image library.

$CCVJ008 Skeleton

Skeleton JCL for printing reports in batch. The symbolics for this skeleton are:

- TAG1—PARM options.
- TAG3—LST class, number of copies, print destination.

$CCVJ009 Skeleton

Skeleton JCL for automatic job checkoff and EOJ processing.

Skeleton JCL step for automatic job checkoff. The symbolic for this skeleton is:

- TAG11—CCF-generated job number.

$CCVJ010 Skeleton

Skeleton JCL for production source assembly. The symbolics for this skeleton are:

- TAG0—CA Librarian program name.
- TAG1—Member name.
- TAG2—Production master file
- TAG4—Production master file MCD.
- TAG7—Production master volser.
- TAG12—Production member password.
- TAG15—Production relocatable library.

$CCVJ011 Skeleton

Skeleton JCL for production source COBOL compile. The symbolics for this skeleton are:

- TAG1—Member name.
- TAG2—Production master file.
- TAG4—Production master file MCD.
- TAG7—Production master volser.
- TAG12—Production member password.
- TAG15—Production relocatable library.

$CCVJ012 Skeleton

Skeleton JCL for production source PL1 compile. The symbolics for this skeleton are:

- TAG0—CA Librarian program name.
- TAG1—Member name.
- TAG2—Production master file.
- TAG4—Production master file MCD.
- TAG7—Production master volser.
- TAG12—Production member password.
- TAG15—Production relocatable library.

$CCVJT10 Skeleton

Skeleton JCL for test source assembly. The symbolics for this skeleton are:

- TAG0—CA Librarian program name.
- TAG1—Member name.
- TAG2—Production master file.
- TAG3—Test master file.
- TAG4—Production master file MCD.
- TAG5—Test master file MCD.
- TAG7—Production master volser.
- TAG8—Test master volser.
- TAG9—Production master filetype and catname.

- TAG10—Test master filetype and catname.
- TAG12—Test member password.
- TAG15—Production relocatable library.
- TAG16—Test relocatable library.

$CCVJT11 Skeleton

Skeleton JCL for test source COBOL compile. The symbolics for this skeleton are:

- TAG0—CA Librarian program name.
- TAG1—Member name.
- TAG2—Production master file.
- TAG3—Test master file.
- TAG4—Production master file MCD.
- TAG5—Test master file MCD.
- TAG7—Production master volser.
- TAG8—Test master volser.
- TAG9—Production master filetype and catname.
- TAG10—Test master filetype and catname.
- TAG12—Test member password.
- TAG15—Production relocatable library.
- TAG16—Test relocatable library.

$CCVJT12 Skeleton

Skeleton JCL for test source PL1 compile. The symbolics for this skeleton are:

- TAG0—CA Librarian program name.
- TAG1—Member name.
- TAG2—Production master file.
- TAG3—Test master file.
- TAG4—Production master file MCD.
- TAG5—Test master file MCD.
- TAG7—Production master volser.

- TAG8—Test master volser.
- TAG9—Production master filetype and catname.
- TAG10—Test master filetype and catname.
- TAG12—Test member password.
- TAG15—Production relocatable library.
- TAG16—Test relocatable library.

$CCVJ090 Skeleton

Skeleton JCL for production archive index option.

Skeleton JCL for archive index report. The symbolics for this skeleton are:

- TAG1—Member name.
- TAG2—Production master file.
- TAG4—Production master file MCD.
- TAG7—Production master volser.
- TAG9—Production master filetype, catalog name.

$CCVJ100 Skeleton

Skeleton JCL for test/production compare option. The symbolics for this skeleton are:

- TAG0—CA Librarian batch program name.
- TAG1—Member name.
- TAG2—Production master file.
- TAG3—Test master file.
- TAG4—Production master file MCD.
- TAG5—Test master file MCD.
- TAG6—Current member version.
- TAG7—Production master file volser.
- TAG8—Test master file volser.
- TAG12—Test module password.
- TAG13—Production module password.

$CCVJ200 Skeleton

Skeleton JCL for Advantage CA-MetaCOBOL+.

Manipulating JCL Processing

This section describes how you can alter LIB/CCF JCL processing and sequences of events.

As supplied on the installation tape, the JCL skeletons perform certain functions in a certain order. The JCL skeletons are stored as CA Vollie VOICE library members in the OLL where LIB/CCF was installed. LIB/CCF builds job streams from several different skeletons by copying each appropriate skeleton into a member named CCFJOB. By definition, these skeletons have symbolics that you must replace with actual values before they can be used as JCL. Once all the needed skeletons are copied into CCFJOB, LIB/CCF performs an EDIT of the symbolics, changing them to their assigned values.

For example, consider how LIB/CCF prepares the JCL for a login using LIBDCOPY. First, it copies skeleton $CCVJ004, which has three parts, to member CCFJOB. The first part is labeled GETTEST (the first half of LIBDCOPY), the second part is labeled PUTPROD (the second half of LIBDCOPY), and the third part is labeled DELTEST, to delete the member from the test master file.

LIB/CCF next copies the appropriate JCL compile skeleton, which it determines by obtaining the language type for the member from the History Master file, and then using it to search the LIB/CCF Language Definition Table. There are two parts to a compile skeleton.

- The first selects the member from the production master file and writes it to JOBSTR.
- The second part compiles the member. Then it copies skeleton $CCVJ000 to the CCFJOB, which contains the job card. Finally, LIB/CCF appends the JSR Checkoff step, skeleton $CCVJ009, to the very end. The EDIT to translate the symbolics is performed, and then CCFJOB is submitted.

Once you understand the mechanics of how LIB/CCF builds JCL streams, it is clear that by manipulating the contents of the individual skeletons, you can build job streams in any way. The following sections examine two common scenarios.

Compile from the Test Master File

During login, some sites would like the member compiled before it is moved from test to production. This means compiling from the test master file. You can do this with some minor changes to the JCL skeleton $CCVJ004 and the compile skeleton for the appropriate language type. This example uses the assembler skeleton, $CCVJ010.

Skeleton $CCVJ004 has three logical parts to it, steps GETTEST, PUTPROD, and DELTEST. Each of these steps must execute after the compile is performed. Move these steps to the end of skeleton $CCVJ010, following the assembly step, leaving at least one comment statement in $CCVJ004. The first part of $CCVJ010 selects the member from the production master file. Change the symbolics TAG2 and TAG7 to TAG3 and TAG8 respectively to have it select the member from the test master file instead.

You must make these changes for each language type that requires this processing.

Compile and Link at Login

Because many programs consist of only one member, you often want to compile and link at login instead of in separate steps. This saves the programmer the extra step of submitting a System Link Request. You can do this with modifications to the compile JCL skeleton for each individual language type.

Again using the assembler example, append a LNKEDT step to the assembly step in skeleton $CCVJ010. The skeleton now has three steps: select, assembly, and link. On the LIBDEF statement in the link step, use the symbolic TAG8 for the core image library.

In the LIB/CCF Master File Table Processor (Option 12.4), whatever is specified as the RELATED PRODUCTION RELOCATABLE library is substituted in the JCL skeleton for the TAG8 symbolic. Therefore, specify the core image library name in this field. As a result, you lose the production relocatable library needed in the MAINT step that follows the assembly. Bypass the problem by removing the MAINT step.

Keep in mind that, since the link-edit is included in the login step, the programmer can no longer execute the LIB/CCF System Link Request (Option 6). As a result, the programmer is not prompted for link-edit options like AUTO, and so on, at the time the link-edit is submitted. They must be hard coded into the skeleton.

Link-Edit Considerations

Two Option 12.0 (PROFILE) keywords, CLINK and LKED, affect how link-edits are performed in LIB/CCF. See the System Profile Table secion in the “Administrative Functions in LIB/CCF” chapter for details on each keyword.

From Option 6 (LINK), the programmer is prompted for a Change Request ID and then sees a list of production master files and associated core image libraries. He must then enter the number of the pair that defines the master file containing the associated linkage editor input and the core image library where the phases should be linked.

The programmer must also enter either a LINK INPUT MEMBER or a LINK JCL MEMBER. These fields are mutually exclusive. Information about both fields follows:

LINK INPUT MEMBER

This field:

- Must reside on the production master file.
- Must contain the required link-edit PHASE and INCLUDE statements. Other link-edit control statements are optional.
- Specifying this field displays the Linkage Editor Options Selection panel. To choose an option, place an **X** next to the display. Selected options are included on the generated link-edit ACTION control statement.
- Automatic phase backup (when requested) is based on the names specified in the PHASE statements.

LINK JCL MEMBER

This field:

- Must reside in the CA Vollie library where LIB/CCF is installed.
- Must contain the complete link-edit JCL stream, must consist of a single job, and include POWER JCL.
- The member is submitted as is.
- Automatic phase backup is not performed unless a step is inserted into the JCL stream to accomplish it.

Note: The sample user exits described earlier in this chapter allow the use of a member name variable (@MODNAME@) in place of the actual member name in the LINK INPUT MEMBER and LINK JCL MEMBER. See the section titled Sample User Exits earlier in this chapter for details on exits $CCVU001 and $CCVU008.

Customizing Panels

After installing LIB/CCF, you can customize some of the panels to reflect information that is more important to your site. The following sections describe the panels and reports that you can modify.

The following is a list of the VOICE procs containing panels LIB/CCF uses. A more detailed description of each of these panels follows in the next section.

$CCVP000

Primary LIB/CCF panel. This is the panel on which you select the LIB/CCF functions. You can modify this panel to remove options that your site does not use.

- The label is PAN0.
- The associated help panel is $CCVH005.

$CCVP001

Panel to open a change request. You can change the data area definitions on this panel. You must also make these changes in the panels contained in $CCVP002 and $CCVP003. Also change the report formatter service program $CCVS015.

- The label is PAN1.
- The associated help panel is $CCVH010.

$CCVP002

Panel to assign a change request. You can change the data area definitions on this panel. Also change the panels contained in $CCVP001 and $CCVP003.

- The label is PAN2.
- The associated help panel is $CCVH020

$CCVP003

Panel to view a change request. You can change the data area definitions on this panel. Also change the panels contained in $CCVP001 and $CCVP002.

- The label is PAN3.
- The associated help panel is $CCVH030.

$CCVP007

Panel to close a change request. Function program $CCVP007 controls processing for this panel. The report formatter service program $CCVS017 must also reflect the changes. This proc contains all three panels for the CLOSE function.

- The labels are PAN7, PANB, and PANC.
- The associated help panel is $CCVH070.

$CCVP008

Panel to select a report generator. You can modify this panel to remove or add additional reports.

- The panel label is PAN8.
- The associated help panel is #CCVH080.

$CCVS007

Panel for history member creation. You cannot modify or move the data field for the LANG. You can modify the other data description fields of this panel. Also change the report formatter service program $CCVS022.

- The panel label is PAND.
- The associated help panel is n/a.

$CCVS008

Panel for history member update. You can modify the data description fields of this panel. Make the same changes to the report formatter service program $CCVS022.

- The panel label is PANE.
- The associated help panel is n/a.

$CCVS031

Panels for LIB/CCF system error notification. Modify these panels to reflect the telephone number of the LIB/CCF system administrator.

- The panel labels are .SENDPAN and ERRORPAN
- The associated help panel is $CCVH008.

$CCVS041

Panel for link options specification. You can modify this panel to specify different default options.

- The panel label is PN00.
- The associated help panel is n/a.

Customizing Reports

LIB/CCF is distributed with a CA Earl (Easy Access Report Language) interface. CA Earl is a report generation system that uses a high-level programming language to produce customized reports. The LIB/CCF-Advantage CA-Earl interface provides access to information LIB/CCF collected and stored on the System Master file. See Chapter 11, “LIB/CCF-Advantage CA-Earl Interface,” for details.

In addition to using the LIB/CCF-Advantage CA-Earl interface to generate reports, you can modify the online reports available from Option 8 (REPORTS). Listed below are the LIB/CCF reports that you can modify and where they are created.

Module History Report

This report is formatted by service program $CCVS022. Modify it if you modified either of the panels in $CCVS007 or $CCVS008. The report formatting program inserts title lines into the history data at the appropriate locations. If you changed just the data descriptions, then these title lines are changed. If you added new data descriptions, then insert the title lines.

Change Request Report

The report driver service program, $CCVS018, controls the creation of this report. The report is composed of three parts:

- The description section, which is formatted by service program $CCVS015.
- The activity section, which is formatted by service program $CCVS016
- And the CLOSE section, which is formatted by service program $CCVS017.

Do not modify the activity section. Modify the description section if you changed the description panel in $CCVP001. Modify the close section if you changed any of the panels in $CCVP007.

The report formatting program inserts title lines into the Change Request data at the appropriate locations. If you changed only the data descriptions, then these title lines are changed. If you added new data descriptions, insert the title lines.

The program (CCFB200) that prints online print requests, when PRINT BATCH is specified in Option 12.0, recognizes the following ANS control characters: ‘1 (eject) and 0 (skip 1). Any line already formatted must contain a tilde or an accent grave in position 1. Otherwise, it is formatted nine spaces to the right.

Execution

After LIB/CCF is installed, you can execute it by entering the name of the entry program: %CCF. Optionally, you can make an option specification as follows: %CCF 3 or %CCF 8.2

PF Keys and Commands

PF key definitions for the LIB/CCF system are non-modifiable and are set as follows:

- PF1 and PF13 (- HELP)
- PF3 and PF15 (- END)
- PF5 and PF17 (- NEXT)
- PF7 and PF19 (- SCROLL BACKWARD)
- PF8 and PF20 (- SCROLL FORWARD)

- PF10 and PF22 (- TOP)
- PF11 and PF23 (- BOTTOM)

A jump command enables a direct jump to a specified option. The syntax is:

```
=option<.option>
```

Examples: =3 or =8.5

HELP Displays HELP information for the current panel.

Job Card Specification

The JCL job statement skeleton is used when LIB/CCF jobs are submitted for batch execution. Use Option 0 to appropriately modify the job statement to reflect the correct information. Once a job is submitted from the LIB/CCF user's key, an CA Vollie member is created in the user's CA Vollie library containing the job card to use for subsequent jobs. The AllFusion CA-Vollie member name is $CCVJ000. There are two variables available from Option 0 that are filled in when a batch job is submitted from LIB/CCF. Use of these variables is optional. They are:

JNAME

LIB/CCF generates a jobname of *opident*CCF, where *opident* is the user's CA Vollie OPIDENT.

PNAME

The CA Vollie user key.

The job statement can consist of up to ten lines and must begin on the first line. Any trailing unused lines are ignored.

System Error Processing

For CA Vollie CCF, the errors can be severe. Severe errors are those that only the LIB/CCF administrator can resolve. When such an error occurs, LIB/CCF sends a message to the LIB/CCF administrator's CA Vollie library as member CCFERROR.

The message is placed on the SYSTEM OLL.

Should a LIB/CCF-AllFusion CA-Vollie system error occur, an error panel displays showing the program and function where the error occurred. The LIB/CCF system administrator is notified of the error automatically. If you encounter such an error, do not try to continue using LIB/CCF without contacting the administrator and resolving the problem.

The error panel displays a message if an error occurs while an attempt is made to notify the administrator. If this happens, relay the information on the error panel to the administrator. For all other errors, LIB/CCF creates the messages as a member in the user's CA Vollie library.

Appendix A: Extension Variables

The following is a list of variables that you can use in LIB/CCF, z/OS and OS/390, and VM/ESA JCL skeletons. The substituted value for each variable is obtained from the Library Chain Definition Function (LCDF) except where noted. All variables obtained from LCDF list an LCDF ORIGIN, denoting an LCDF field. See the Library Chain Definition Function section in the “Administrator Functions in LIB/CCF” chapter for a detailed explanation of the fields. We use the terms TO and FROM throughout the list to refer to target and sending libraries, respectively. The variables are available to any LIB/CCF batch job (with the exception of print jobs).

```
-------------------------------------------------------------
ROSCOE   ISPF       LGTH. DESCRIPTION
VARIABLE VARIABLE
NAME     NAME
-------------------------------------------------------------
LIBRARY TYPE - LCDF ORIGIN: TYPE
-------------------------------------------------------------
TAGX001  &ITF        1    FROM LIBRARY TYPE
TAGX002  &ITT        1    TO LIBRARY TYPE
                          INDICATORS:
                          'P' FOR PROD LIBRARY
                          'T' FOR TEST LIBRARY
                          'R' FOR REJECT LIBRARY
                          'Q' FOR Q/A LIBRARY

-------------------------------------------------------------
HOW TO MOVE SOURCE - LCDF ORIGIN: SOURCE MOVE
-------------------------------------------------------------
TAGX003  &ISM        1    INDICATORS:
                          'C' MOVE CURRENT ARCHIVE LEVEL
                          'A' MOVE ALL ARCHIVE LEVELS

-------------------------------------------------------------
LOCK STATUS AFTER MOVE - LCDF ORIGIN: STATUS
-------------------------------------------------------------
TAGX004  &ILOCK          1    INDICATORS:
                              'T' FOR TEST
                              '0' FOR PROD0
                              '1' FOR PROD1
                              '2' FOR PROD2

-------------------------------------------------------------
Appendix A. Extension Variables A-1
-------------------------------------------------------------
ROSCOE   ISPF       LGTH. DESCRIPTION
VARIABLE VARIABLE
NAME     NAME

-------------------------------------------------------------
HOW TO MOVE OBJECT - LCDF ORIGIN: OP SYS MOVE
-------------------------------------------------------------
TAGX005  &IMOVEO     1    INDICATORS:
                          'N' FOR DO NOT MOVE OBJECT
                          'M' FOR MOVE OBJECT
                          'C' FOR COMPILE

-------------------------------------------------------------
HOW TO MOVE LOAD - LCDF ORIGIN: OP SYS MOVE
-------------------------------------------------------------
```

```
TAGX006  &IMOVEL          INDICATORS:
                          'N' DO NOT MOVE LOAD
                          'M' MOVE LOAD

-------------------------------------------------------------
WHERE TO MOVE FROM REJECT - LCDF ORIGIN: PROMOTE
-------------------------------------------------------------
TAGX007  &IREJP             1    INDICATORS:
                                 'B' MOVE TO Q/A LIBRARY
                                     FROM WHICH IT WAS REJECTED
                                 'F' MOVE TO THE FIRST Q/A
                                     LIBRARY IN THE CHAIN

-------------------------------------------------------------
LOCK STATUS WITH NO MOVE - LCDF ORIGIN: LOCK
-------------------------------------------------------------
TAGX008  &ILOCKT           1    INDICATORS:
                                 'T' FOR TEST
                                 '0' FOR PROD0
                                 '1' FOR PROD1
                                 '2' FOR PROD2

-------------------------------------------------------------
MOVE FROM TEST AT LOGIN - LCDF ORIGIN: LOGIN MOVE
-------------------------------------------------------------
TAGX009   &ILOGINDE       1    INDICATORS:
                                 'Y' MOVE AT LOGIN
                                 'N' DO NOT MOVE AT LOGIN

-------------------------------------------------------------
A-2 CA-Librarian 4.3 LIB/CCF Implementation Guide
-------------------------------------------------------------
ROSCOE   ISPF           LGTH. DESCRIPTION
VARIABLE VARIABLE
NAME     NAME

-------------------------------------------------------------
MASTER FILE ACCESS - LCDF ORIGIN: ACCESS
-------------------------------------------------------------
TAGX010  &IPACC      1    THE PROD MASTER ACCESS
TAGX011  &IFACC      1    THE FROM MASTER ACCESS
TAGX012  &ITACC      1    THE TO MASTER ACCESS
TAGX013  &IHACC      1    THE HISTORY MASTER ACCESS
                          INDICATORS:
                          'N' NO ACCESS EXCEPT THROUGH CCF
                          'B' BROWSE ACCESS
                          'R' READ, BROWSE, COPY ACCESS
                          'A' ANY ACCESS
```

```
---------------------------------------------------------------
MCD CODES - LCDF ORIGIN: MCD BASE
            NOTE: THESE VARIABLES WILL BE SUBSTITUTED WITH
            THE CORRECT MCD (I.E., THE DATE PLUS THE MCD
            BASE FROM LCDF).
---------------------------------------------------------------
TAGX014   &IPMCD     4     THE PROD MASTER MCD
TAGX015   &IFMCD     4     THE FROM MASTER MCD
TAGX016   &ITMCD     4     THE TO MASTER MCD
TAGX017   &IHMCD     4     THE HISTORY MASTER MCD

---------------------------------------------------------------
MASTER FILE DATA SET NAMES - LCDF ORIGIN: DATA SET NAME
---------------------------------------------------------------
TAGX018     &IPMAST     44     THE PROD MASTER
TAGX019     &IFMAST     44     THE FROM MASTER
TAGX020     &ITMAST     44     THE TO MASTER
TAGX021     &IHMAST     44     THE HISTORY MASTER

---------------------------------------------------------------
Appendix A. Extension Variables A-3
---------------------------------------------------------------
ROSCOE   ISPF           LGTH. DESCRIPTION
VARIABLE VARIABLE
NAME     NAME
---------------------------------------------------------------
MVS LIBRARIES - LCDF ORIGIN FOR EACH LIBRARY IS:
                OBJECT LIBRARY
                LOAD LIBRARY
                DBRM LIBRARY
                BACKUP LIBRARY
---------------------------------------------------------------
TAGX022  &IMPO      44   MVS PROD OBJECT LIBRARY
TAGX023  &IMPOBU    44   MVS PROD OBJECT LIBRARY (BACKUP)
TAGX024  &IMPL      44   MVS PROD LOAD LIBRARY
TAGX025  &IMPLBU    44   MVS PROD LOAD LIBRARY (BACKUP)
TAGX026  &IMPD      44   MVS PROD DBRM LIBRARY
TAGX027  &IMPDBU    44   MVS PROD DBRM LIBRARY (BACKUP)
TAGX028  &IMFO      44   MVS FROM OBJECT LIBRARY
TAGX029  &IMFOBU    44   MVS FROM OBJECT LIBRARY (BACKUP)
TAGX030  &IMFL      44   MVS FROM LOAD LIBRARY
TAGX031  &IMFLBU    44   MVS FROM LOAD LIBRARY (BACKUP)
TAGX032  &IMFD      44   MVS FROM DBRM LIBRARY
TAGX033  &IMFDBU    44   MVS FROM DBRM LIBRARY (BACKUP)
TAGX034  &IMTO      44   MVS TO OBJECT LIBRARY
TAGX035  &IMTOBU    44   MVS TO OBJECT LIBRARY (BACKUP)
TAGX036  &IMTL      44   MVS TO LOAD LIBRARY
TAGX037  &IMTLBU    44   MVS TO LOAD LIBRARY (BACKUP)
TAGX038  &IMTD      44   MVS TO DBRM LIBRARY
```

```
TAGX039  &IMTDBU    44   MVS TO DBRM LIBRARY (BACKUP)

-------------------------------------------------------------
MISCELLANEOUS - NOT OBTAINED FROM LCDF
-------------------------------------------------------------
TAGX000  &$CCFDB2    4    DB2 SUBSYSTEM NAME
TAGX100  &ITYPE      1    JOB TYPE INDICATOR
                          'I' LOGIN
                          'O' LOGOUT
                          'L' LINK OR DB2 BIND
                          'R' REJECT
TAGX101  &WO         8    CHANGE REQUEST NUMBER
TAGX102  &CCFID      8    CCF ID (ACTUAL USERID OR CONVERTED
                          CCF ID FROM OPTION 12.8)

-------------------------------------------------------------
A-4 CA-Librarian 4.3 LIB/CCF Implementation Guide
-------------------------------------------------------------
          ISPF LGTH.           DESCRIPTION
          VARIABLE
          NAME

-------------------------------------------------------------
THE FOLLOWING CMS AND VSE VARIABLES ARE AVAILABLE ONLY TO
LIB/CCF CMS:
-------------------------------------------------------------
CMS MASTER FILE INFORMATION - LCDF ORIGIN FOR EACH VARIABLE:
                              TYPE
                              OWNER
                              ADDRESS
                              MULTIWRITE PASSWORD
-------------------------------------------------------------
          &ICPLTYPE   1   CMS PROD MASTER TYPE
          &ICFLTYPE   1   CMS FROM MASTER TYPE
          &ICTLTYPE   1   CMS TO MASTER TYPE
          &ICHLTYPE   1   CMS HIST MASTER TYPE
                          INDICATORS:
                          'D' BDAM FORMAT
                          'C' CMS FORMAT

          &ICPMOWN    8   CMS PROD MASTER OWNER
          &ICPMAD     4   CMS PROD MASTER ADDRESS
          &ICPMPW     8   CMS PROD MASTER PASSWORD
          &ICFMOWN    8   CMS FROM MASTER OWNER
          &ICFMAD     4   CMS FROM MASTER ADDRESS
          &ICFMPW     8   CMS FROM MASTER PASSWORD
          &ICTMOWN    8   CMS TO MASTER OWNER
          &ICTMAD     4   CMS TO MASTER ADDRESS
          &ICTMPW     8   CMS TO MASTER PASSWORD
```

```
          &ICHMOWN    8   CMS HIST MASTER OWNER
          &ICHMAD     4   CMS HIST MASTER ADDRESS
          &ICHMPW     8   CMS HIST MASTER PASSWORD

-------------------------------------------------------------
Appendix A. Extension Variables A-5
-------------------------------------------------------------
          ISPF            LGTH. DESCRIPTION
          VARIABLE
          NAME

-------------------------------------------------------------
CMS OBJECT LIBRARY INFORMATION - LCDF ORIGIN FOR EACH VARIABLE:
                                OWNER
                                ADDR
                                PASSWORD
                                FILE NAME
                                FILE TYPE
                                MODE
-------------------------------------------------------------
          &ICPOOWN    8   CMS PROD OBJECT OWNER
          &ICPOAD     4   CMS PROD OBJECT ADDRESS
          &ICPOPW     8   CMS PROD OBJECT PASSWORD
          &ICPOFN     8   CMS PROD OBJECT FILE NAME
          &ICPOFT     8   CMS PROD OBJECT FILE TYPE
          &ICPOLM     2   CMS PROD OBJECT LINK MODE
          &ICPOOWNB   8   CMS PROD OBJECT OWNER (BACKUP)
          &ICPOADB    4   CMS PROD OBJECT ADDRESS (BACKUP)
          &ICPOPWB    8   CMS PROD OBJECT PASSWORD (BACKUP)
          &ICPOFNB    8   CMS PROD OBJECT FILE NAME (BACKUP)
          &ICPOFTB    8   CMS PROD OBJECT FILE TYPE (BACKUP)
          &ICPOLMB    2   CMS PROD OBJECT LINK MODE (BACKUP)
          &ICFOOWN    8   CMS FROM OBJECT OWNER
          &ICFOAD     4   CMS FROM OBJECT ADDRESS
          &ICFOPW     8   CMS FROM OBJECT PASSWORD
          &ICFOFN     8   CMS FROM OBJECT FILE NAME
          &ICFOFT     8   CMS FROM OBJECT FILE TYPE
          &ICFOLM     2   CMS FROM OBJECT LINK MODE
          &ICFOOWNB   8   CMS FROM OBJECT OWNER (BACKUP)
          &ICFOADB    4   CMS FROM OBJECT ADDRESS (BACKUP)
          &ICFOPWB    8   CMS FROM OBJECT PASSWORD (BACKUP)
          &ICFOFNB    8   CMS FROM OBJECT FILE NAME (BACKUP)
          &ICFOFTB    8   CMS FROM OBJECT FILE TYPE (BACKUP)
          &ICFOLMB    2   CMS FROM OBJECT LINK MODE (BACKUP)
          &ICTOOWN    8   CMS TO OBJECT OWNER
          &ICTOAD     4   CMS TO OBJECT ADDRESS
          &ICTOPW     8   CMS TO OBJECT PASSWORD
          &ICTOFN     8   CMS TO OBJECT FILE NAME
          &ICTOFT     8   CMS TO OBJECT FILE TYPE
```

```
          &ICTOLM     2   CMS TO OBJECT LINK MODE
          &ICTOOWNB   8   CMS TO OBJECT OWNER (BACKUP)
          &ICTOADB    4   CMS TO OBJECT ADDRESS (BACKUP)
          &ICTOPWB    8   CMS TO OBJECT PASSWORD (BACKUP)
          &ICTOFNB    8   CMS TO OBJECT FILE NAME (BACKUP)
          &ICTOFTB    8   CMS TO OBJECT FILE TYPE (BACKUP)
          &ICTOLMB    2   CMS TO OBJECT LINK MODE (BACKUP)

-------------------------------------------------------------
A-6 CA-Librarian 4.3 LIB/CCF Implementation Guide
-------------------------------------------------------------
          ISPF              LGTH. DESCRIPTION
          VARIABLE
          NAME
-------------------------------------------------------------
CMS LOAD LIBRARY INFORMATION - LCDF ORIGIN FOR EACH VARIABLE:
                               OWNER
                               ADDR
                               PASSWORD
                               FILE NAME
                               FILE TYPE
                               MODE
-------------------------------------------------------------
          &ICPLOWN    8    CMS PROD LOAD OWNER
          &ICPLAD     4    CMS PROD LOAD ADDRESS
          &ICPLPW     8    CMS PROD LOAD PASSWORD
          &ICPLFN     8    CMS PROD LOAD FILE NAME
          &ICPLFT     8    CMS PROD LOAD FILE TYPE
          &ICPLLM     2    CMS PROD LOAD LINK MODE
          &ICPLOWNB   8    CMS PROD LOAD OWNER (BACKUP)
          &ICPLADB    4    CMS PROD LOAD ADDRESS (BACKUP)
          &ICPLPWB    8    CMS PROD LOAD PASSWORD (BACKUP)
          &ICPLFNB    8    CMS PROD LOAD FILE NAME (BACKUP)
          &ICPLFTB    8    CMS PROD LOAD FILE TYPE (BACKUP)
          &ICPLLMB    2    CMS PROD LOAD LINK MODE (BACKUP)
          &ICFLOWN    8    CMS FROM LOAD OWNER
          &ICFLAD     4    CMS FROM LOAD ADDRESS
          &ICFLPW     8    CMS FROM LOAD PASSWORD
          &ICFLFN     8    CMS FROM LOAD FILE NAME
          &ICFLFT     8    CMS FROM LOAD FILE TYPE
          &ICFLLM     2    CMS FROM LOAD LINK MODE
          &ICFLOWNB   8    CMS FROM LOAD OWNER (BACKUP)
          &ICFLADB    4    CMS FROM LOAD ADDRESS (BACKUP)
          &ICFLPWB    8    CMS FROM LOAD PASSWORD (BACKUP)
          &ICFLFNB    8    CMS FROM LOAD FILE NAME (BACKUP)
          &ICFLFTB    8    CMS FROM LOAD FILE TYPE (BACKUP)
          &ICFLLMB    2    CMS FROM LOAD LINK MODE (BACKUP)
          &ICTLOWN    8    CMS TO LOAD OWNER
          &ICTLAD     4    CMS TO LOAD ADDRESS
```

```
          &ICTLPW     8    CMS TO LOAD PASSWORD
          &ICTLFN     8    CMS TO LOAD FILE NAME
          &ICTLFT     8    CMS TO LOAD FILE TYPE
          &ICTLLM     2    CMS TO LOAD LINK MODE
          &ICTLOWNB   8    CMS TO LOAD OWNER (BACKUP)
          &ICTLADB    4    CMS TO LOAD ADDRESS (BACKUP)
          &ICTLPWB    8    CMS TO LOAD PASSWORD (BACKUP)
          &ICTLFNB    8    CMS TO LOAD FILE NAME (BACKUP)
          &ICTLFTB    8    CMS TO LOAD FILE TYPE (BACKUP)
          &ICTLLMB    2    CMS TO LOAD LINK MODE (BACKUP)

-------------------------------------------------------------
Appendix A. Extension Variables A-7
-------------------------------------------------------------
          ISPF           LGTH. DESCRIPTION
          VARIABLE
          NAME

-------------------------------------------------------------
VSE MASTER FILE VOLUME INFORMATION - LCDF ORIGIN: VOLUME ID
-------------------------------------------------------------
          &IVPMVOL    6    VSE PROD MASTER VOLUME
          &IVFMVOL    6    VSE FROM MASTER VOLUME
          &IVTMVOL    6    VSE TO MASTER VOLUME
          &IVHMVOL    6    VSE HIST MASTER VOLUME

-------------------------------------------------------------
VSE OBJECT LIBRARIES - LCDF ORIGIN FOR EACH VARIABLE:
                       LIBRARY or SUBLIBRARY
-------------------------------------------------------------
          &IVPOL    7    VSE PROD OBJECT LIBRARY
          &IVPOS    8    VSE PROD OBJECT SUBLIBRARY
          &IVPOLB   7    VSE PROD OBJECT LIBRARY (BACKUP)
          &IVPOSB   8    VSE PROD OBJECT SUBLIBRARY (BACKUP)

          &IVFOL    7    VSE FROM OBJECT LIBRARY
          &IVFOS    8    VSE FROM OBJECT SUBLIBRARY
          &IVFOLB   7    VSE FROM OBJECT LIBRARY (BACKUP)
          &IVFOSB   8    VSE FROM OBJECT SUBLIBRARY (BACKUP)

          &IVTOL    7    VSE TO OBJECT LIBRARY
          &IVTOS    8    VSE TO OBJECT SUBLIBRARY
          &IVTOLB   7    VSE TO OBJECT LIBRARY (BACKUP)
          &IVTOSB   8    VSE TO OBJECT SUBLIBRARY (BACKUP)

-------------------------------------------------------------
VSE PHASE LIBRARIES - LCDF ORIGIN FOR EACH VARIABLE:
                      LIBRARY or SUBLIBRARY
-------------------------------------------------------------
```

```
                &IVPLL     7    VSE PROD PHASE LIBRARY
                &IVPLS     8    VSE PROD PHASE SUBLIBRARY
                &IVPLLB    7    VSE PROD PHASE LIBRARY (BACKUP)
                &IVPLSB    8    VSE PROD PHASE SUBLIBRARY (BACKUP)

                &IVFLL     7    VSE FROM PHASE LIBRARY
                &IVFLS     8    VSE FROM PHASE SUBLIBRARY
                &IVFLLB    7    VSE FROM PHASE LIBRARY (BACKUP)
                &IVFLSB    8    VSE FROM PHASE SUBLIBRARY (BACKUP)

                &IVTLL     7    VSE TO PHASE LIBRARY
                &IVTLS     8    VSE TO PHASE SUBLIBRARY
                &IVTLLB    7    VSE TO PHASE LIBRARY (BACKUP)
                &IVTLSB    8    VSE TO PHASE SUBLIBRARY (BACKUP)

------------------------------------------------------------------
```

Appendix B: Troubleshooting

This chapter contains information about:

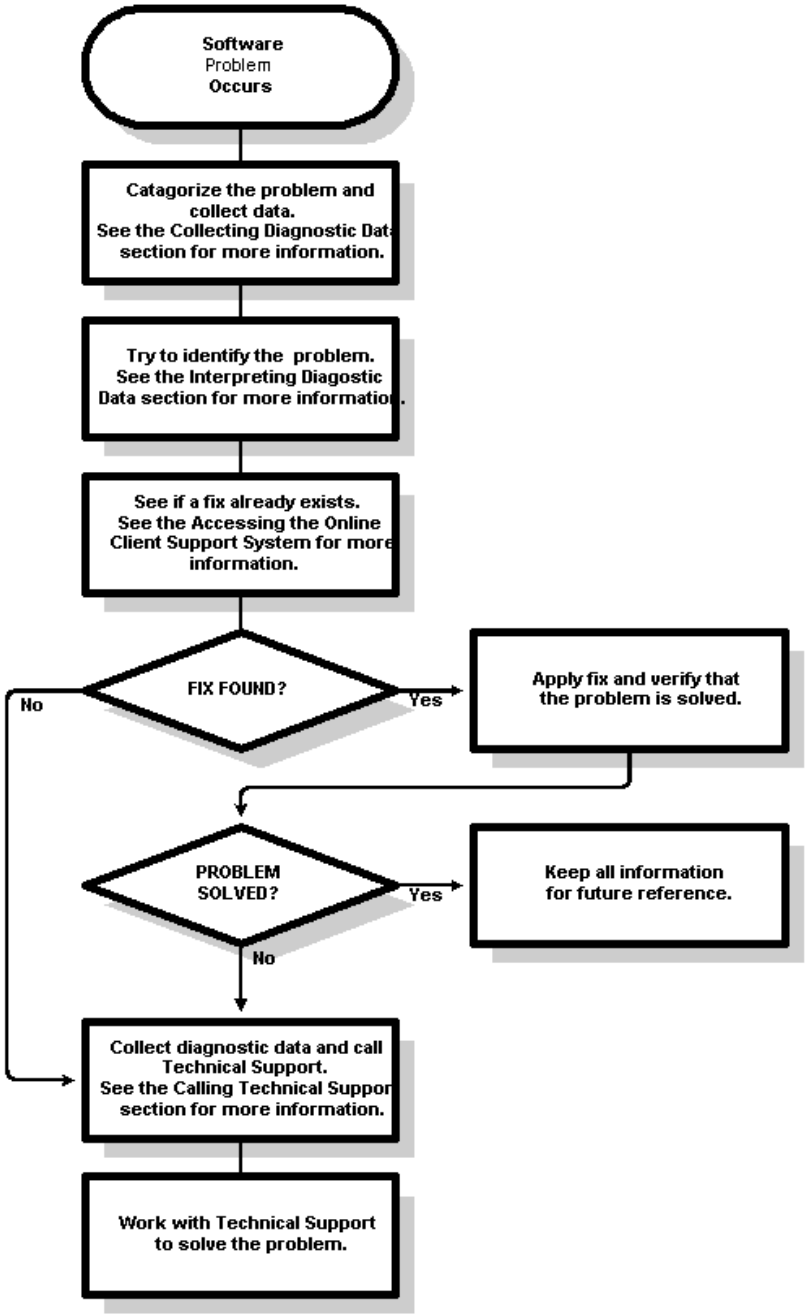
- Identifying and resolving problems
- Contacting CA Technical Support
- Receiving a new version of a product and ongoing maintenance
- Requesting product enhancements

This section contains the following topics:

Diagnostic Procedures (see page 334)
Collecting Diagnostic Data (see page 335)
Interpreting Diagnostic Data (see page 335)
Accessing the Online Client Support System (see page 336)
Accessing the Technical Support Phone Services Directory (see page 338)
CA-TLC: Total License Care (see page 338)
Contacting Technical Support (see page 339)
Product Versions and Maintenance (see page 339)
Requesting Enhancements (see page 340)

Diagnostic Procedures

See the flowchart below for a summary of the procedures you should follow if you have a problem with a CA software product. Each of these procedures is detailed on the following pages.



Collecting Diagnostic Data

The following information is helpful in diagnosing problems that might occur:

- Control statements used to activate your product
- JCL used to install or activate your product
- Relevant system log or console listings
- Relevant system dumps or product dumps
- List of other IBM or third-party products that might be involved
- Manufacturer, model number, and capacity of your hardware
- Numbers and text of IBM or CA error messages associated with the problem
- Names of panels where the problem occurs
- Listings of all fixes applied to all relevant software, including:
  - The dates fixes were applied
  - Fix numbers
  - Names of components to which fixes were applied
- Short description of problems

Interpreting Diagnostic Data

When you have collected the specified diagnostic data, write down your answers to the following questions:

1. What was the sequence of events prior to the error condition?

   What circumstances existed when the problem occurred and what action did you take?
2. Has this situation occurred before? What was different then?
3. Did the problem occur after a particular PTF was applied or after a new version of the software was installed?
4. Have you recently installed a new version of the operating system?
5. Has the hardware configuration (tape drives, disk drives, and so forth) changed?

From your response to these questions and the diagnostic data, try to identify the cause and resolve the problem.

Accessing the Online Client Support System

CA is making extensive use of the Internet for your benefit. CA encourages you to "surf the net" to the CA home page at http://www.ca.com and the support site at http://support.ca.com. The CA Internet site provides a great variety of information about CA products and services, including:

- Service and support
- Product information and sales
- CA-World conference information
- Press releases
- CA user groups

StarTCC

StarTCC, the web-based portion of CA-Total Client Care (CA-TCC), gives you real time, interactive access to CA product support information through the Internet. Using StarTCC, you can:

- Open new issues
- Browse or update your existing issues and enhancement requests
- Perform keyword searches
- Download solutions, PTFs, and important notices regarding

CA products, maintenance, and documentation

Requirements for Using StarTCC

The following are the requirements to use StarTCC:

- You must be a CA client with a current maintenance agreement.
- You must register through the CA Internet site.
- You must access the Internet with a browser that supports the HTML specification 2.0 or higher, such as Netscape Navigator 2.0 or higher or Microsoft Internet Explorer 3.0 or higher. Browsers that meet the HTML requirement support the following functions, which are required for StarTCC:
  - Secure sockets layer (SSL) to encrypt your transaction traffic
  - Encrypted data records (known as COOKIES)
  - HTML tables

StarTCC Security

StarTCC runs as a secured server (SSL). You may need to configure your browser to enable SSL. Guidelines for doing this are provided on the CA Technical Support page.

Accessing StarTCC

To access StarTCC, go to eSupport.ca.com The StarTCC options are:

- StarTCC Information
- StarTCC Registration
- Access StarTCC

These options are described below.

StarTCC Information

Select the information option to view background information for StarTCC, details about the prerequisites, and instructions for configuring your browser. Be sure to review this section for updates or information not included here.

StarTCC Registration

Select the registration option to identify yourself to StarTCC. You must register before you can access StarTCC online. There are prompts for all required information, including your name, site ID, CA-StarTrak PIN, company name, E-Mail address, postal address, and desired password for accessing StarTCC.

Note: If you do not have a CA-StarTrak PIN, StarTCC provides one for you when you register.

Access StarTCC

Select the access option to begin using StarTCC. When prompted, enter your user ID and password. Once your sign-on is validated, you can perform the following:

Open a New Issue

Open an issue for, or request an enhancement to, one of your CA products.

Browse Your Issues And Enhancement Requests

Display all issues for your site. The issues are grouped into three categories: Open, Closed, and Enhancement Requests (DARs).

Browse and/or Download Solutions

Specify criteria for selecting solutions, which you can then view or download.

Search the CA Knowledge Base

Specify criteria for searching the CA database for solutions, problems, and keywords that can provide you with immediate answers to your product support questions and concerns.

Update Your StarTCC Profile

Make changes to your default E-mail address, phone number, and password whenever necessary.

Display Your Site's Licenses

View a list of all the CA products for which your company site is currently licensed.

Display StarTCC News Items

View and download recently published solutions for CA products, instructions for downloading from StarTCC, and helpful information for using CA-StarTrak, StarTCC, or other CA products.

Accessing the Technical Support Phone Services Directory

The CA Technical Support Phone Services Directory lists each CA product and the telephone number to call for primary support for that product. To access the Support Phone Services Directory, set your browser for http://support.ca.com and click on Contact Us.

CA-TLC: Total License Care

Many CA software solutions use license keys or authorization codes to validate your hardware configuration. If you need assistance obtaining a license key or authorization code, contact the CA-TLC: Total License Care group through http://support.ca.com.

Contacting Technical Support

For further technical assistance with this product, please contact CA Technical Support at http://support.ca.com for a complete list of CA locations and phone numbers.

Technical Support is available 24 hours a day, seven days a week.

If you are unable to resolve the problem, please have the following information ready before contacting CA Technical Support:

- All the diagnostic information described in "Collecting Diagnostic Data" on page 6-4
- Product name, version number, operating system, and genlevel.
- Product name and version number of any other software you suspect is involved.
- Version level and PUTLEVEL of the operating system.
- Your name, telephone number and extension (if any).
- Your company name.
- Your site ID.
- A severity code. This is a number (from 1 to 4) that you assign to the problem. Use the following to determine the severity of the problem:
  - 1 indicates a "system down" or inoperative condition.
  - 2 indicates a suspected high-impact condition associated with the product.
  - 3 indicates a question concerning product performance or an intermittent low-impact condition associated with the product.
  - 4 indicates a question concerning general product utilization or implementation.

Product Versions and Maintenance

Clients are requested to operate only under currently supported versions of the product.

Clients with current maintenance agreements also receive ongoing product maintenance. When a new version of the system is available, a notice is sent to all current clients.

Requesting Enhancements

CA welcomes your suggestions for product enhancements. All suggestions are considered and acknowledged.

You can use either of two methods to request enhancements:

- Contact your Account Manager who will initiate a Demand Analysis Request (DAR) for you.
- Enter your request through StarTCC, the CA web-based, interactive support system at http://support.ca.com.

Appendix C: Reserve/Release Test Programs

The correct functioning of the RESERVE/RELEASE commands in a user's environment is critical to data integrity of a master file. With this in mind, there are three programs to test the reserve/release commands at your site. We recommend you run these tests, though they are not required. A VSE/ESA-only site need not run the programs.

The z/OS and OS/390 load module is named LVRTESTO and is link edited during the installation of the z/OS and OS/390 batch CA Librarian.

The VSE/ESA phase is named LIBRRES and is link edited during the installation of the VSE/ESA batch CA Librarian. The four-character prefix of the phase might be changed during the installation.

The VM/ESA and z/VM module is named LVRTESTC and is created during the LIB/CMS installation.

This section contains the following topics:

How the Programs Work (see page 341)
Testing Reserve/Release From z/OS and OS/390 (see page 342)
Testing Reserve/Release From VSE/ESA (see page 343)
Testing Reserve/Release From VM/ESA and z/VM (see page 346)

How the Programs Work

A pair of programs running on separate machines (real or virtual) verify that RESERVE/RELEASE is functioning properly between the two machines. To insure the integrity of a master file, all environments where VM/ESA and z/VM or z/OS and OS/390 are accessing a master file must have RESERVE/RELEASE functioning properly.

One program initializes a one track file. In the case where two VM/ESA and z/VM machines are checked, this first step must be accomplished in a separate run on an z/OS and OS/390 or VSE/ESA batch machine.

Then the pair of programs synchronizes with each other.

One machine RESERVEs the disk. The other verifies that the RESERVE is functioning.

Then the first machine RELEASEs the disk and the other RESERVEs it.

In all tests, you must specify one machine as the FIRST and the other as the SECOND. In addition, you can run Z/OS and OS/390 and VSE/ESA to INITIALIZE the file for a subsequent VM/ESA and z/VM run.

In Z/OS and OS/390, the PARM field indicates what type of run is required. The values for the PARM field are FIRST, SECOND, and INITIALIZE.

In VSE/ESA, the run type is indicated by the use of UPSI flags. A flag of:

- 1XX indicates FIRST.
- X1X indicates SECOND.
- XX1 indicates INITIALIZE.
- XXX1 indicates that an FBA device is being used.

In VM/ESA and z/VM, a PARM of FIRST or SECOND is used.

Note: All of the LVRTEST messages are documented in the *Messages Guide.* The message LVRTEST RESERVE FROM OTHER MACHINE FAILED indicates that these programs were able to access a file that should be protected with a RESERVE. Therefore, an integrity exposure exists. Consult your systems programmer to correct the access configuration to this file. This problem must be corrected before accessing the master file from this environment.

The LVRTEST RUN COMPLETE message indicates that the

RESERVE/RELEASE mechanism is functioning correctly.

Testing Reserve/Release From z/OS and OS/390

The following sample JCL is required to run the RESERVE/RELEASE tests on an z/OS and OS/390 machine.

```
// jobname JOB
   column=42./stepname EXEC PGM=LVRTEST0,PARM='parm'            A
// SYSPRINT DD SYSOUT=*
   column=52./SYSUT1 DD DSN=file.name,DISP=(NEW,CATLG,CATLG),   B
// DCB=DSORG=PS,VOL=SER=disk,
   column=39./ SPACE=(100,4),UNIT=DISK                          C
//
```

The following explains values that appear in the sample JCL:

A

The PARM is required, and must be one of the following:

- FIRST—This is the first program. It initializes the file, synchronizes it with the second program, reserves, and checks that the other program reserves properly. If z/OS and OS/390 and VSE/ESA are being tested, z/OS and OS/390 must be the FIRST.
- SECOND—This is the second program. It must be started after the first program initialized the file. It synchronizes with the first program, checks that the first program reserved properly, and reserves the file.
- INITIALIZE—This program only initializes the file for later use by two VM/ESA and z/VM programs.

B

The value of *file.name* must be the same for both programs being tested.

C

Enough space to create 4 100-byte records is required.

Testing Reserve/Release From VSE/ESA

The following JCL is required to run the RESERVE/RELEASE tests on a VSE/ESA machine using a CKD device for the test file.

```
// JOB jobname
// DLBL SYSUT1,'file name',,DA          A
// EXTENT SYS774                        B
// DLBL SYSUT2,'file name',,SD          C
// EXTENT SYS774,volser,1,0,nnn,1       D
// ASSGN SYS774,DISK,VOL=volser,SHR     E
// UPSI xxx                             F
// EXEC LIBRRES,SIZE=AUTO               G
 /*
 /&
```

The following explains values that appear in the sample JCL:

A

SYSUT1 is the ddname of the file to reserve. This statement is always required. The file name for this file must match the one for SYSUT2 in the job that creates the file.

B

An extent for SYS004 is required.

C

SYSUT2 is the ddname of the file to create. It is required only in jobs that are creating the file. Its file name must match the file name in SYSUT1.

D

The extent must be on SYS004 and be large enough to hold 4 100-byte records (one track).

E

SYS004 must be assigned to the disk drive holding the test file. SYS004 must be the logical unit number and it must be used in all extents.

F

The UPSI is required and has the following meanings:

- 1XX—This is the first program. It initializes the file, synchronizes it with the second program, reserves, and checks that the other program reserves properly.
- X1X—This is the second program. It must be started after the first program initialized the file. It synchronizes with the first program, checks that the first program reserved properly, and reserves the file.XX1—This program only initializes the file for later use by two VM/ESA and z/VM programs.

Note: Any other UPSI is invalid.

G

The execute should specify a size of AUTO. The phase name for the VSE/ESA reserve/release verification program is the first four characters specified in the VSE/ESA LIBGEN macro with the suffix of RES added.

The following JCL is required to run the RESERVE/RELEASE tests on a VSE/ESA machine using a FBA device for the test file.

```
// JOB jobname
// DLBL SYSFBA1,'filename1'            A
// EXTENT SYS774                      B
// DLBL SYSFBA2,'filename1'            C
// EXTENT SYS774,volser,1,0,nn,4      D
// ASSGN SYS774,DISK,VOL=volser,SHR   E
// UPSI xxx1                          F
// EXEC LIBRRES,SIZE=AUTO             G
 /*
 /&
```

The following explains values that appear in the sample JCL:

A

SYSFBA1 is the ddname of the file on an FBA device to reserve. This statement is always required. This file name must match the one for SYSFBA2 in the job that creates the file.

B

An extent for SYS004 is required.

C

SYSFBA2 is the ddname of the file to create on an FBA device. It is required only in jobs that are creating the file. Its file name must match the file name in SYSFBA1.

D

The extent must be on SYS004 and be large enough to hold four FBA blocks (one track or four FBA blocks).

E

SYS004 must be assigned to the disk drive holding the test file. SYS004 must be the logical unit number and it must be used in all extents.

F

The UPSI is required and has the following meanings:

- 1XX—This is the first program. It initializes the file, synchronizes it with the second program, reserves, and checks that the other program reserves properly.
- X1X—This is the second program. It must be started after the first program initialized the file. It synchronizes with the first program, checks that the first program reserved properly, and reserves the file.
- XX1—This program only initializes the file for later use by two VM/ESA and z/VM programs.
- XXX1—FBA device is used for the test file.

Note: Any other UPSI is invalid.

G

The execute should specify a size of AUTO. The phase name for the VSE/ESA reserve/release verification program is the first four characters specified in the VSE/ESA LIBGEN macro with the suffix of RES added.

Testing Reserve/Release From VM/ESA and z/VM

The LIB/CMS command LVRTESTC is provided for RESERVE/RELEASE testing. The following example shows how to execute the testing program on a VM/ESA and z/VM machine:

```
FILEDEF RSYSUT1 filemode DSN data set name    A
LVRTESTC SECOND starting block number         B
```

The following explains values that appear in the sample JCL:

A

The VM/ESA and z/VM command FILEDEF with ddname RSYSUT1 defines the file to test the proper RESERVE/RELEASE configuration. The file must be the one initialized by either the z/OS and OS/390 or VSE/ESA program.

B

The command LVRTESTC is invoked using the parameter SECOND. The command must be issued after the first program from Z/OS and OS/390 or VSE/ESA starts. While the first program initialized the file and is waiting for the second program to start to synchronize, the command synchronizes with the first program. Check that the first program reserved properly. Information is provided in the typeouts. If the file resides on an FBA type DASD, the command needs the starting block number of the file. You can specify it as the second parameter of the command, as follows: LVRTESTC SECOND STARTING-BLOCK-NUMBER

You can obtain the starting block number of the file by issuing the VM/ESA and z/VM command LISTDS with EXTENT option.

Glossary

application

An *application* is a user-defined name used to refer collectively to any group of programs performing related functions. Authorized User Key Definition Table defining to LIB/CCF the application users who can initiate change requests and the applications for which they can initiate requests.

backup load library

A *backup load library* is an optional library where executable production programs can be backed up before a new program is link edited. There must be one for each production library to back up.

CA Librarian

CA Librarian is a ,CA Technologies library management system. This system consists of a specially formatted, self-reorganizing library and the batch utility programs and online interfaces that access it.

CA Librarian Change Control Facility (LIB/CCF)

The *CA Librarian Change Control Facility (LIB/CCF)* is an interactive dialog-based application that provides CA Librarian users with a comprehensive application change control methodology. It includes complete tracking, reporting, and control of the program development and maintenance process in online environments.

CA Roscoe key conversion table

[set the rie variable for your book] key conversion table defines LIB/CCF users' [set the rie variable for your book] keys and the corresponding LIB/CCF keys to the LIB/CCF-[set the rie variable for your book] system.

CA Vollie OPNAME conversion table

CA Vollie OPNAME conversion table defines LIB/CCF users' CICS operator names and the corresponding CCF ID.

CCF administrator

A *CCF administrator* is an individual designated by site management as having responsibility for implementing LIB/CCF and maintaining its associated tables. This person establishes the LIB/CCF system profile, defines the users of the system, and establishes their authority to exercise specific LIB/CCF functions.

change request

A *change request* is a request for a modification or problem resolution. The change request is used as an anchor point for all LIB/CCF functions.

change request ID

A *change request ID* is a change request number.

Close function

Close function removes a change request from the list of active or open requests.

CMS userid conversion table

The *CMS userid conversion table* defines LIB/CCF users' VM/ESA user IDs and the corresponding LIB/CCF keys to the LIB/CCF system.

control group

A *control group* contains the user IDs, defined by the LIB/CCF system administrator, responsible for processing programmer requests (logout, login, link, and DB2 bind requests). Additionally, the control group can be responsible for processing job submission records (JSRs).

History master file definition table

A *History master file definition table* defines to LIB/CCF the History master file associated with each Production master file.

History member

A *History member* is a member on the History master file that contains a cumulative narrative describing all changes ever made to the corresponding source module in the production master file.

job submission record (JSR)

A *job submission record (JSR)* is a record created when a batch job is submitted to perform a login, logout, link-edit, or DB2 bind of a module.

Jump command

The *Jump command* is a command consisting of an equal sign and panel number (=n) that lets the online user display another panel without passing through a hierarchy.

LANGUAGE definition table

The *LANGUAGE definition table* is a table of LIB/CCF language names and the JCL members set up to compile programs written in the corresponding programming language.

LIB/CCF system master

LIB/CCF system master is a CA Librarian master file designated as the repository of LIB/CCF system information. This information includes all change requests, member tracking information, and general system status information.

library chain

A *library chain* is a group of CA Librarian master files that define a promotion path for a member. A library chain consists of at least a Production and Test master file and can include any number of intermediate (QA or Reject) master files.

Library Chain Definition function (LCDF)

Library Chain Definition Function (LCDF) is a LIB/CCF administrator function for defining Production, Test, and intermediate (QA and Reject) master files, their associated libraries (object, load, history), and other related information.

library definition record (LDR)

A *library definition record (LDR)* is a record created for each master file or operating system library defined by the administrator with LCDF. An LDR contains all information needed by LIB/CCF to access the file, such as data set name, VM/ESA link options, master file type, and so on.

Link function

The *Link function* initiates the process of link editing a member that was updated and returned to the production master file.

Login function

The *Login function* is a LIB/CCF function that processes a request to move a copy of a new or changed program back into production. The login can be processed directly or proceed in steps that involve movement of the program to an intermediate library for quality assurance purposes.

Logout function

The *Logout function* is a LIB/CCF function that processes a request to move a copy of a production program to a designated test library so that changes can be made to it in response to a change request.

logout stamp

A *logout stamp* indicates the date and time of the logout.

master file definition table

A *master file definition table* defines the production master file to the LIB/CCF system.

model system

A *model system* is a generalized LIB/CCF system supplied by CA, tailored to no particular site.

module movement record (MMR)

A *module movement record* (MMR) is a record created by LIB/CCF for use by the control group to manage requests from programmers to transfer a member.

module tracking record (MTR)

A *module tracking record (MTR)* is a record created by LIB/CCF when a member is logged out of the production environment to the test environment.

production link request

A *production link request* is a request to link edit a member in a production master file.

production master file

A *production master file* is a CA Librarian master file defined to LIB/CCF as a repository of production source code. When a change is required to a production program, a request is made to LIB/CCF to move a copy of that member to the development environment.

production object library

A *production object library* is a library that is the target of production compiles or where the test object member is copied if a TEST OBJECT is specified; the library where the object members are link edited.

programmer definition table

A *programmer definition table* defines programmers to the LIB/CCF system.

programming managers definition table

A *programming managers definition table* defines programming managers to the LIB/CCF system.

system link request (SLR)

A *system link request (SLR)* is a LIB/CCF function that relinks an entire system or subsystem after one of its parts is changed. A link request can only be processed after login is completed successfully.

test master file

A *test master file* is a CA Librarian master file defined to LIB/CCF as the library where copies of production programs reside during the development phase.

test object/relocatable

A *test object/relocatable* is an optional object code representing the output from the most recent compile of the test member being logged in can be copied from this library into the production object library.

TSO userID conversion table

A *TSO userID conversion table* is a table that defines LIB/CCF users' TSO user IDs and the corresponding LIB/CCF keys to the LIB/CCF-ISPF(TSO) system.

Index

$

$CCCJ200 EXEC skeleton • 275
$CCCJ200 JCL skeleton • 275
$CCCJ201 EXEC skeleton • 275
$CCCJ201 JCL skeleton • 275
$CCCTnnn
    system master member • 120
$CCFB101 (Change Request Status Report) • 79
$CCFB102 (Batch Logout Utility) • 81
$CCFB102 condition codes • 81
$CCFB109 (Batch Login Utility) • 84
$CCFGEN macro • 113
$CCFJ000 EXEC skeleton • 275
$CCFJ000 JCL skeleton • 244
$CCFJ001 JCL skeleton • 244
$CCFJ002 JCL skeleton • 244
$CCFJ003 JCL skeleton • 244
$CCFJ004 JCL skeleton • 244
$CCFJ005 JCL skeleton • 244
$CCFJ006 JCL skeleton • 244
$CCFJ007 JCL skeleton • 244
$CCFJ008 JCL skeleton • 244
$CCFJ009 JCL skeleton • 244
$CCFJ010 JCL skeleton • 244
$CCFJ011 JCL skeleton • 244
$CCFJ012 JCL skeleton • 244
$CCFJ030 EXEC skeleton • 275
$CCFJ030 JCL skeleton • 244, 275
$CCFJ031 EXEC skeleton • 275
$CCFJ031 JCL skeleton • 244, 275
$CCFJ032 EXEC skeleton • 275
$CCFJ032 JCL skeleton • 244, 275
$CCFJ040 JCL skeleton • 244
$CCFJ100 JCL skeleton • 244
$CCFJ101 JCL skeleton • 244
$CCFJ102 JCL skeleton • 244
$CCFJ103 EXEC skeleton • 275
$CCFJ103 JCL skeleton • 244, 275
$CCFJ200 EXEC skeleton • 275
$CCFJ200 JCL skeleton • 244, 275
$CCFJ201 EXEC skeleton • 275
$CCFJ201 JCL skeleton • 244, 275
$CCFJ500 JCL skeleton • 244
$CCFJ501 JCL skeleton • 244
$CCFJT10 JCL skeleton • 244
$CCFJT11 JCL skeleton • 244
$CCFJT12 JCL skeleton • 244
$CCFPN00 panel • 292
$CCFPN08 panel • 292
$CCFPN18 panel • 292
$CCFTnnn
    system master member • 120
$CCFU000 user exit • 231, 263, 273
$CCFU001 user exit • 231, 263, 273
$CCFU002 • 273
$CCFU002 user exit • 231, 263
$CCFU003 user exit • 231, 263, 273
$CCFU004 user exit • 231, 263, 273
$CCFU005 user exit • 231, 263, 273
$CCFU006 user exit • 231, 263
$CCFU007 sample user exit • 242
$CCFU007 user exit • 231, 263, 273
$CCFU008 user exit • 231, 263, 273
$CCFU009 user exit • 231, 263, 273
$CCFU010 user exit • 231, 263
$CCFU012 user exit • 231, 263
$CCFU013 user exit • 231, 263
$CCFU014 user exit • 231
$CCFU015 user exit • 231, 263, 273, 305
$CCRFB104 (Batch Change Request Close) • 97
$CCVJ000 JCL skeleton • 307
$CCVJ001 JCL skeleton • 307
$CCVJ002 JCL skeleton • 307
$CCVJ003 JCL skeleton • 307
$CCVJ004 JCL skeleton • 307
$CCVJ005 JCL skeleton • 307
$CCVJ006 JCL skeleton • 307
$CCVJ007 JCL skeleton • 307
$CCVJ008 JCL skeleton • 307
$CCVJ009 JCL skeleton • 307
$CCVJ010 JCL skeleton • 307
$CCVJ011 JCL skeleton • 307
$CCVJ012 JCL skeleton • 307
$CCVJ090 JCL skeleton • 307
$CCVJ100 JCL skeleton • 307
$CCVJ200 JCL skeleton • 307
$CCVJT10 JCL skeleton • 307
$CCVJT11 JCL skeleton • 307
$CCVJT12 JCL skeleton • 307

$CCVP000 panel • 316
$CCVP001 panel • 316
$CCVP002 panel • 316
$CCVP003 panel • 316
$CCVP007 panel • 316
$CCVP008 panel • 316
$CCVS007 panel • 316
$CCVS008 panel • 316
$CCVS031 panel • 316
$CCVS041 panel • 316
$CCVTnnn
system master member • 120
$CCVU000 user exit • 297, 305
$CCVU001 user exit • 297, 305
$CCVU002 user exit • 297, 305
$CCVU003 user exit • 297, 305
$CCVU004 user exit • 297, 305
$CCVU005 user exit • 297, 305
$CCVU006 user exit • 297
$CCVU007 user exit • 297, 305
$CCVU008 user exit • 297, 305
$CCVU010 user exit • 297, 305
$CCVU011 user exit • 297, 305
$CCVU015 user exit • 297
$CCxJ000 JCL skeleton • 275
$CCxJ001 EXEC skeleton • 275
$CCxJ001 JCL skeleton • 275
$CCxJ002 EXEC skeleton • 275
$CCxJ002 JCL skeleton • 275
$CCxJ003 EXEC skeleton • 275
$CCxJ003 JCL skeleton • 275
$CCxJ004 EXEC skeleton • 275
$CCxJ004 JCL skeleton • 275
$CCxJ005 EXEC skeleton • 275
$CCxJ005 JCL skeleton • 275
$CCxJ006 EXEC skeleton • 275
$CCxJ006 JCL skeleton • 275
$CCxJ007 EXEC skeleton • 275
$CCxJ007 JCL skeleton • 275
$CCxJ009 EXEC skeleton • 275
$CCxJ009 JCL skeleton • 275
$CCxJ010 EXEC skeleton • 275
$CCxJ010 JCL skeleton • 275
$CCxJ011 EXEC skeleton • 275
$CCxJ011 JCL skeleton • 275
$CCxJ012 EXEC skeleton • 275
$CCxJ012 JCL skeleton • 275
$CCxJ100 EXEC skeleton • 275
$CCxJ100 JCL skeleton • 275
$CCxJ101 EXEC skeleton • 275
$CCxJ101 JCL skeleton • 275
$CCxJT10 JCL skeleton • 275
$CCxJT11 JCL skeleton • 275
$CCxJT12 JCL skeleton • 275
$CHGJ000 JCL skeleton • 207
$CHGJ001 JCL skeleton • 207
$CHGJ002 JCL skeleton • 207
$CHGJ003 JCL skeleton • 207
$CHGJ004 JCL skeleton • 207
$CHGJ005 JCL skeleton • 207
$CHGJ006 JCL skeleton • 207
$CHGJ007 JCL skeleton • 207
$CHGJ008 JCL skeleton • 207
$CHGJ009 JCL skeleton • 207
$CHGJ010 JCL skeleton • 207
$CHGJ011 JCL skeleton • 207
$CHGJ012 JCL skeleton • 207
$CHGJ030 JCL skeleton • 207
$CHGJ031 JCL skeleton • 207
$CHGJ032 JCL skeleton • 207
$CHGJ040 JCL skeleton • 207
$CHGJ100 JCL skeleton • 207
$CHGJ102 JCL skeleton • 207
$CHGJ103 JCL skeleton • 207
$CHGJ200 JCL skeleton • 207
$CHGJ201 JCL skeleton • 207
$CHGJ500 JCL skeleton • 207
$CHGJ501 JCL skeleton • 207
$CHGJT10 JCL skeleton • 207
$CHGJT11 JCL skeleton • 207
$CHGJT12 JCL skeleton • 207
$CHGTnnn
system master member • 120
$CHGU000 user exit • 200, 206
$CHGU001 user exit • 200, 206
$CHGU002 user exit • 200, 206
$CHGU003 user exit • 200, 206
$CHGU004 user exit • 200, 206
$CHGU005 user exit • 200, 206
$CHGU006 user exit • 200, 206
$CHGU007 user exit • 200, 206
$CHGU008 user exit • 200, 206
$CHGU010 user exit • 206
$CHGU011 user exit • 200, 206
$CHGU012 user exit • 200
$CHGU013 user exit • 200
$CHGU014 user exit • 200, 206
$CHGU015 user exit • 200, 206

%

% variable • 229

)

)CM processing statement • 218
)ENDSEL processing statement • 218
)IM member processing statement • 218
)SEL <expression> processing statement • 218
)TRACE processing statement • 218

/

// ID control statement • 106

[

[OPEN|UNASSIGNED|CLOSED|ALL|CRID=] parameter • 184
[UNCONDITIONAL|CONDITIONAL] parameter • 82, 92, 97, 106, 109, 113

=

=n (jump) command • 37, 38, 40, 42, 46, 48, 50, 51, 53, 54, 55, 56, 58, 77

0

0 condition code • 81, 85

2

20 codition code • 85

4

4 condition code • 81

8

8 condition code • 81, 85

A

-AATH code control statement • 109
-ABST [text of abstract record] control statement • 106
-ABST[text of abstract record] SYSIN Control Statement • 82
ACCESS field • 43, 70
ACTION field • 61
ACTIVITY parameter • 184
ADD/DELETE LIBRARY MODIFIER field • 70
ADDR field • 72
ADDRESS field • 72
ADMINID keyword • 22
ADMINKEY keyword • 22
Advantage CA-Roscoe key conversion table • 52
-AFFT list control statement • 98, 113
ALL field • 38
ALL parameter • 80
AllFusion CA-Vollie OPNAME Conversion Table panel • 54
ALLOWMULT keyword • 22
ALLOWMULT YES • 129
altddname parameter • 96, 100
-ANAL analysis control statement • 98, 113
ANALYSIS parameter • 80, 104, 184
-ANME ccfid control statement • 109
APP • 180
-APPL [appl sys] control statement • 106
-APPL appl sys control statement • 109
-APPL[appl sys] SYSIN Control Statement • 82
APPL=[application name] parameter • 80, 104
APPL=<application name> parameter • 184
APPLICATION field • 39
-ASGN [WOnnnnnn] processing verb • 110
Associated Files panel • 66, 73
Associated Files panel fields • 66
audit trail for DB2 for z/OS and OS/390 • 175
AUTHORIZATION CODE field • 38, 39
AUTHORIZED APPLICATION field • 38
AUTOREPT1 keyword • 22
AUTOREPT2 keyword • 22
AUTOREPT5 keyword • 22
AUTOREPT7 keyword • 22

B

BACK command • 59
BACKUP LIBRARY field • 70
BASE field • 57
Batch Change Request Close ($CCFB104) • 97
Batch Change Request Close parameter • 97
Batch Change Request Close program • 113
  example of • 115
  executing from another program • 115
  parameters for • 113
  processing verb • 114
  specifications for • 115
  SYSIPT control statements • 113
Batch Change Request Initiation parameters • 92
Batch Change Request Initiation program • 108

example of • 111
executing from another program • 112
parameters for • 109
processing verbs for • 110
specifications for • 111
SYSIPT control statements for • 109
Batch Change Request Initiation program ($CCFB103) • 92
Batch Login Utility ($CCFB109) • 84
Batch Login Utility parameters • 85
Batch Login Utility SYSIN Control Statements • 86
Batch Logout utility • 105
example of • 108
parameters for • 106
sample JCL • 84
specifications for • 107
SYSIPT control statements for • 106
Batch Logout Utility ($CCFB102) • 81
Batch Logout Utility condition codes • 81, 85
Batch Logout Utility parameters • 82
Batch Logout Utility SYSIN control statements • 82
BATCHCMS keyword • 22
BATCHMVS keyword • 22
BATCHVSE keyword • 22
BKUPLOAD • 122

C

CA7JCLID keyword • 22, 166
CA7OPERID keyword • 22, 166
CA7PASS keyword • 22, 166
CAEARL EXEC • 191
CAlIBCCF.LIBCCFTA table, maintaining • 182
CALIBCCF.LIBCCFTA table, updating • 180
CANCEL command • 37, 38, 40, 42, 46, 48, 50, 51, 53, 54, 55, 58, 59, 77
CCF ID field • 54
CCF KEY field • 53, 55
CCF USER ID field • 70
CCFB103 (Batch Change Request Initiation program • 92
CCFEARL MACLIB library • 191
CCFEARL TXTLIB library • 191
CCFENQUE
system master member • 120
CCFEXIT EXEC • 191
CCFID userid SYSIN control statements • 86
Chain Attributes panel • 63, 73
Chain Attributes panel fields • 63
Chain Description panel • 61, 73
CHAIN GROUP field • 39, 43, 60
CHAIN MODEL field • 60
CHAIN NUMBER field • 60
Chain Selection panel • 73
Chain Structure panel • 61, 73
Chain Structure panel fields • 61
change control facility (LIB/CCF) • 13
change request • 118
Change Request Reassignment panel • 55
Change Request Reassignment panel commands • 56
Change Request Reassignment panel fields • 56
Change Request report • 319
Change Request Status report • 103
JCL for • 103
parameters for • 104
Change Request Status Report ($CCFB101) • 79
Change Request Status Report parameter • 80
CLINK keyword • 22, 166
CLOSE keyword • 22
CLOSE WOnnnnnn processing verb • 98
-CLOSE WOnnnnnn processing verb • 114
CLOSED parameter • 80
CMOVEMENT keyword • 22, 166
CNTLID keyword • 22
CNTLKEY keyword • 22
CNTLPRINT keyword • 22
-COM [comment] control statement • 106
-COM comment parameter • 114
-COM comment processing verb • 98, 110
-COM[comment] SYSIN Control Statement • 82
-COMM comment control statement • 109
commands
Change Request Reassignment panel commands • 56
group definition table panel commands • 77
ISPF SYSUID Conversion Table panel commands • 54
Language Definition Table panel commands • 48
LCDF panel commands • 59
Manager Definition Table panel commands • 40
Master File Information Table panel commands • 58
OPNAME Conversion Table panel commands • 55
Programmer Definition Table panel commands • 42
System Profile Table panel commands • 37

User Definition Table panel commands • 38
compile and link member at login • 223
compile and link members at login • 315
compile and link the member at login • 255
compile member from the test master file • 222
compile members before moving • 314
compile the member from the test master file • 255
condition codes
$CCFB102 • 81
Batch Logout Utility condition codes • 85
conditional JCL skeleton generation • 218
control statements
Batch Change Request Close program control statements • 113
Batch Change Request Initiation program SYSIPT control statements • 109
Batch Logout utility control statements • 106
Batch Logout Utility SYSIN control statements • 82
SYSIN control statements • 98
-COPY [modname] control statement • 106
-COPY [modname] SYSIN Control Statement • 82
COPY field • 43
-CORD name control statement • 109
-CPH# number control statement • 109
create a library chain • 60
Create Chain panel fields • 60
Create Chain panel sample • 60
-CRID WOnnnnnn control statement • 106
-CRID WOnnnnnn SYSIN Control Statement • 82
cridaddr parameter • 96, 100, 112, 116
CRREPT keyword • 22
-CUID [id] SYSIN Control Statement • 82
CURWONUM
system master member • 120
customize with exit points • 128
customizing panels • 316
customizing panels and reports • 292
customizing reports • 318

D

DATA SET NAME field • 70
DATEFORM keyword • 22
DB2 for z/OS and OS/390 LIB/CCF audit trail • 175
DB2 for z/OS and OS/390 support in LIB/CCF • 171
DB2 SUBSYSTEM field • 295
DB2PKG keyword • 22
DBRM LIBRARY field • 66
DBRMLIB field • 180
DBRMNAME field • 180
DBRMTIME field • 180
DD field • 61
deactivate a library chain • 74
Deactivate Chain panel • 74
DEFAULT SYSTEM field • 295
defaults for model system • 122
delete deactivated chains • 75
demo script for model system • 123
-DESC [desc] control statement • 106
-DESC desc control statement • 109
-DESC[description] SYSIN Control Statement • 82
DESCRIPTION field • 47, 60
DESELECT [library[(member)]] SYSIN control statements • 86
Dnn field • 61
-DOCC list control statement • 98, 113
DSN dsnid SYSIN control statements • 86

E

emergency changes • 128
EMERID keyword • 22
EMERKEY keyword • 22
-EMOD control statement • 106
-EMOD parameter • 114
-EMOD processing verb • 98, 110
-EMOD SYSIN Control Statement • 82
END command • 59
-END control statement • 106
-END parameter • 114
END PF key command • 37, 38, 40, 42, 48, 53, 54, 56, 77
-END processing verb • 98, 110
-END SYSIN Control Statement • 82
ENTRY field • 43, 57
ENTRY TYPE field • 43
examples
Batch Logout utility • 108
EXEC skeletons • 275
EXISTING LIBRARY MODIFIERS field • 70
exit points, customize with • 128
EXITn keyword • 22

F

FCT NAME field • 57
fields
Associated Files panel fields • 66

Chain Attributes panel fields • 63
Chain Structure panel fields • 61
Change Request Reassignment panel fields • 56
Create Chain panel fields • 60
group definition table panel fields • 77
History Master File Definition Table panel fields • 50
ID Conversion Table panel fields • 53
ISPF SYSUID Conversion Table panel fields • 54
Language Definition Table panel fields • 47
Manager Definition Table panel fields • 39
Master File Information Table panel fields • 57
OPNAME Conversion Table panel fields • 55
Production Master File Definition Table panel fields • 49
Programmer Definition Table panel fields • 41
User Definition Table panel fiels • 38
FILE NAME field • 72
FILE ORGANIZATION field • 57
FILE TYPE field • 72
FIND string command • 37, 38, 40, 42, 46, 48, 50, 51, 53, 54, 55, 58, 77
FINISH command • 59
FORMAT keyword • 22
FROM PRODUCTION MASTER FILE-ID field • 43
FROM Q/A MASTER FILE-ID field • 43
function [library[(member)]] SYSIN control statements • 86
function panels, LIB/CCF administrator • 21

G

group definition table • 76
group definition table panel commands • 77
group definition table panel fields • 77
GROUP ID field • 77
Group Processing Activity report • 85
GROUPLOGIN keyword • 22

H

HISTMAST • 122
HISTORY FILE field • 70
History Library panel • 69
History Master File Definition Table panel • 50
History Master File Definition Table panel fields • 50
HISTORY MASTER FILE-ID field • 50
HISTORY update data SYSIN control statements • 86
HISTREPT keyword • 22
HISTUPD keyword • 22

I

ID Conversion Table panel • 52
ID Conversion Table panel fields • 53
identify a LIB/CCF panel • 228
IEXDSLAT • 127
IEXSLAT • 127
implementation considerations (DB2 for z/OS and OS/390) • 173
implementation considerations (job scheduling) • 166
Inn field • 61
-INPT list control statement • 98, 113
installation files for model system • 122
ISPF LIB/CCF primary options menu • 14
ISPF SYSUID Conversion Table panel • 53
ISPF SYSUID Conversion Table panel commands • 54
ISPF SYSUID Conversion Table panel fields • 54

J

JC job statement SYSIN control statements • 86
JCL job statement skeleton • 295
JCL processing
manipulating • 314
JCL processing, manipulating • 217, 254
JCL skeleton, conditional generation • 218
JCL skeletons • 275, 307
JCL skeletons, variables used in • 323
JNAME variable • 229, 320
job card specification • 229, 260, 320
job scheduling for Unicenter CA-7 • 167
job scheduling for Unicenter CA-Scheduler • 168
JOB STATEMENT field • 295
JSQUEUE keyword • 22
JSRCNTL0
system master member • 120
JSREDIT keyword • 22
jump commands • 228, 260, 294

L

-LANG xxxxxxxx control statement • 106
-LANG xxxxxxxx SYSIN Control Statement • 82
Language Definition Table panel • 46
Language Definition Table panel commands • 48
Language Definition Table panel fields • 47
LANGUAGE field • 47
LCDF
Selection List processing • 58

wildcard specification • 58
LCDF panel commands • 59
LCDF screens • 73
LDISPLAY keyword • 22
LIB/CCF
defined • 13
user exits • 297
LIB/CCF administrator • 21
function panels • 21
LIB/CCF basic function • 117
LIB/CCF model system defaults • 122
LIB/CCF request cycle • 14
LIB/CCF-Unicenter CANetman interface
operations • 196
LIBDCOPY utility program • 116
LIBDEXIT • 127
LIBDSCAN • 127
LIBDSCAN utility program • 116
LIBRARIAN MASTER FILE ID field • 57
Library Chain Definition panel • 58
Library Chain Definition panel sample • 59
library chain definition report • 75
sample report • 75
library chain, create a • 60
Library Definition Panels LCDF • 67
LIBRARY field • 61, 63, 72
Library Information panel • 73
Library Selection panel • 73
Library/Chain Status panel • 75
LIBRCOPY utility program • 101
LIBREXIT • 127
LIBRNAME keyword • 22
LIBRSCAN • 100, 127
LINECOUNT keyword • 22
LINES keyword • 22
LINES=nn parameter • 104
LINK INPUT MEMBER field • 315
LINK JCL MEMBER field • 315
link-edit considerations • 224, 256, 290, 315
LJEARLCA EARL • 191
LJEARLCA report generating program • 184
LJEARLCR EARL • 191
LJEARLCR report generating program • 184
LJEARLSY EARL • 191
LJEARLSY report generating program • 185
LKED JCL MEMBER field • 224, 256, 290
LKED keyword • 22, 166
LKED SYSIN MEMBER field • 224, 256, 290
LNKNTEGRTY keyword • 22
LOAD LIBRARY field • 66
LOCK field • 43, 63
LOGIN MOVE field • 63
LOGLIMIT keyword • 22
-LST control statement • 106

M

macros
$CCFGEN • 113
maintaining the CALIBCCF.LIBCCFTA table • 182
manager definition table • 76
Manager Definition Table panel • 39
Manager Definition Table panel commands • 40
Manager Definition Table panel fields • 39
MANAGER field • 56
manipulating JCL processing • 217, 254, 314
Master File Definition Table panel • 42
Master File Definition Table panel fields • 43
Master File Information Table panel • 56
Master File Information Table panel commands • 58
Master File Information Table panel fields • 57
MCD BASE field • 70
MDELETE keyword • 22
MDISPLAY keyword • 22
MEMBER ID field • 77
-MFID nnn control statement • 106
-MFID nnn SYSIN Control Statement • 82
MGR=[managername] parameter • 80, 104
MGR=<managername> parameter • 184
MGRS ID field • 39, 41
MMRCNTL0
system master member • 120
MODE field • 72
model system
defaults • 122
demo script • 123
installation files • 122
requirements • 123
modify the library definition • 73
Module History report • 318
Module Name field • 290
MTRCNTL0
system master member • 120
MULTIWRITE PASSWORD field • 72

N

-NETM id control statement • 109

NEXT command • 37, 38, 40, 42, 46, 48, 50, 51, 53, 55, 77
NOSEP parameter • 80, 104
NOTIFY keyword • 22
NXT parameter • 85

O

-OATH code control statement • 109
OBJECT LIBRARY field • 66
OP SYS MOVE field • 43, 63
-OPEN [WOnnnnnn] processing verb • 110
OPEN parameter • 80
Operating System Libraries panel • 69
operations of LIB/CCF-Unicenter CANetman interface • 196
OPERATOR NAME field • 55
OPNAME Conversion Table panel commands • 55
OPNAME Conversion Table panel fields • 55
optaddr parameter • 96, 100, 112, 116
-OUPT list control statement • 98, 113
OVERRIDE SKELETON field • 224, 256, 290
OVR parameter • 85
OVR0F parameter • 85
OWNER field • 72

P

parameters
  Batch Change Request Close parameters • 97
  Batch Change Request Close program parameters • 113
  Batch Change Request Initiation parameters • 92
  Batch Change Request Initiation program parameters • 109
  Batch Login Utility • 85
  Batch Logout utility parameters • 82, 106
  Change Request Status Report parameters • 80
PASSWORD field • 72
PF keys • 228, 260, 294, 319
PF3/PF15 key command • 46, 50, 51, 55, 58
PGMR ID field • 41
PGMR parameter • 85
PGMR=[pgmrname] parameter • 80, 104
PGMR=<pgmrname> parameter • 184
PMOVEMENT keyword • 22, 166
PNAME variable • 229, 320
POBJECT • 122
PREFIX variable • 229
PRINT keyword • 22
processing statements • 218
processing verb
  Batch Change Request Close program processing verb • 114
processing verbs • 98
  Change Request Initiation program • 110
PROD field • 61
PRODLOAD • 122
PRODMAST • 122
PRODUCTION BACKUP CORE IMAGE LIBRARY field • 49
PRODUCTION CORE IMAGE LIBRARY field • 49
Production Library panel • 67
Production Master File Definition Table panel • 49
Production Master File Definition Table panel fields • 49
PRODUCTION MASTER FILE-ID field • 43, 49, 50
PRODUCTION RELOCATABLE LIBRARY field • 43, 49
PROGR field • 180
programmer definition table • 76
Programmer Definition Table panel • 41
Programmer Definition Table panel commands • 42
Programmer Definition Table panel fields • 41
PROGRAMMER field • 56
programs
  Batch Change Request Close program • 113
  Batch Change Request Initiation program • 108
  Batch Change Request Initiation program ($CCFB103) • 92
  LIBDCOPY utility program • 116
  LIBDSCAN utility program • 116
PROMOTE field • 63
PRTCLASS keyword • 22
PRTY field • 47

Q

Q/A field • 61
Q/A Library panel • 68

R

-RDPT name control statement • 109
REASSIGN TO MANAGER field • 56
REASSIGN TO PROGRAMMER field • 56
recommendations
  security protection • 124
REFRESH SYSIN control statements • 86
REJ field • 61
report generating programs • 184

report generation (VM/ESA) • 191
report generation (VSE/ESA) • 188
reports
  Change Request Status report • 103
  Group Processing Activity • 85
  Pcustomizing • 318
request cycle • 14
REQUEST ID field • 56
requirements for model system • 123
RFIND command • 37, 38, 40, 42, 48, 54, 77
-RMGR name control statement • 109
-RNME name control statement • 109
ROSCOE KEY field • 53
-RPH# number control statement • 109

S

S0nnnnnn
  system master member • 120
sample user exit $CCFU007 • 242
sample user exits • 305
SCHEDTRACE keyword • 22
SCHEDULE [mmddyyhhmmss|*] [predname] SYSIN control statements • 86
SCHEDULER keyword • 22, 166
SCHEDULER keyword operation • 169
scheduling z/OS and OS/390 batch jobs in LIB/CCF • 165
security protection recommendations • 124
Select Library panel • 74
SELECTx WOnnnnnn SYSIN control statements • 86
SKELETON field • 47
SKP parameter • 85
SKP0F parameter • 85
SLRCNTL0
  system master member • 120
SLRCNTL1
  system master member • 120
SORT [GROUP|MEMBER] command • 77
SORT [LANG|TEST|SKEL|PRTY|DESC] command • 48
SORT [MGR|APPL|CHAIN] command • 40
SORT [MGR|PGMR] command • 42
SORT [ROSCOE|CCF] command • 53
SORT [TSO|CMS|CCF] command • 54
SORT [USER|APPL] command • 38
SORT command • 37
SOURCE MOVE field • 63
source to load module verification • 125
specifications
  Batch Change Request Close program • 115
SPOOL MACHINE ID field • 295
SPOOL TAG field • 295
SRCLIB field • 180
SRCNAME field • 180
SRCTIME field • 180
STATUS field • 43, 50, 63
STATUS parameter • 80, 104, 184
SUBLIBRARY field • 72
support in LIB/CCF for DB2 for z/OS and OS/390 • 171
SYS field • 180
SYSIN Control Statements • 98
  Batch Login Utility • 86
SYSIPT control statements
  Batch Logout utility • 106
-SYSM date control statement • 98, 113
SYSMAST • 122
SYSOUT keyword • 22
-SYST list control statement • 98, 113
system error processing • 230, 261, 320
system error processing (LIB/CCF-ISPF(VM/ESA) • 296
SYSTEM LINK EDIT panel • 256
system master files • 120
System Profile Table • 22
  sample screen • 22
System Profile Table panel commands • 37
System Services File
  sample exit programs • 297
SYSTEM=[ISPF|ROSCOE] parameter • 85
SYSTEM=[ISPF|ROSCOE|VOLLIE|CMS] parameter • 82, 92, 109
SYSTEM=[VOLLIE|ISPF|ROSCOE|CMS] parameter • 106
-SYSV date control statement • 98, 113

T

T4nnnnn
  system master member • 120
T5nnnnnn
  system master member • 120
TEST field • 61
TEST MASTER FILE-ID field • 43
TEST RELOCATABLE LIBRARY field • 43
Test/Reject Library panel • 68
TESTMAST • 122
TESTSKEL field • 47

TMOVEMENT keyword • 22, 166
TO Q/A MASTER FILE-ID field • 43
TO REJECT MASTER FILE-ID field • 43
TOTAL ENTRIES field • 43, 57
TSO OR VM/ESA ID field • 54
TYPE field • 61, 70
type parameter • 80, 104
-TYPE[PROBLEM|MODIFICATION] control statement
  • 109

U

UNASSIGNED parameter • 80
Unicenter CA-Scheduler
  job scheduling • 168
Unicenter CA-7
  job scheduling • 167
UNIT • 22
updating the CALIBCCF.LIBCCFTA table • 180
UPPERCASE • 22
-USER ccfid control statement • 98, 109, 113
user definition table • 76
User Definition Table panel commands • 38
User Definition Table panel fields • 38
user exits • 263, 297
  samples • 305
user exits (LIB/CCF-ISPF(TSO)) • 231
USER ID field • 38
USER KEY field • 38
utilities
  Batch Logout utility • 105

V

variables used in JCL skeletons • 323
VM/ESA report generation • 191
VOLSER field • 57
VOLUME ID field • 72
VSAM CATNAME field • 57
VSE/ESA report generation • 188

W

WO field • 180
WOnnnnnn
  system master member • 120

Z

ZRnnnnnn
  system master member • 120