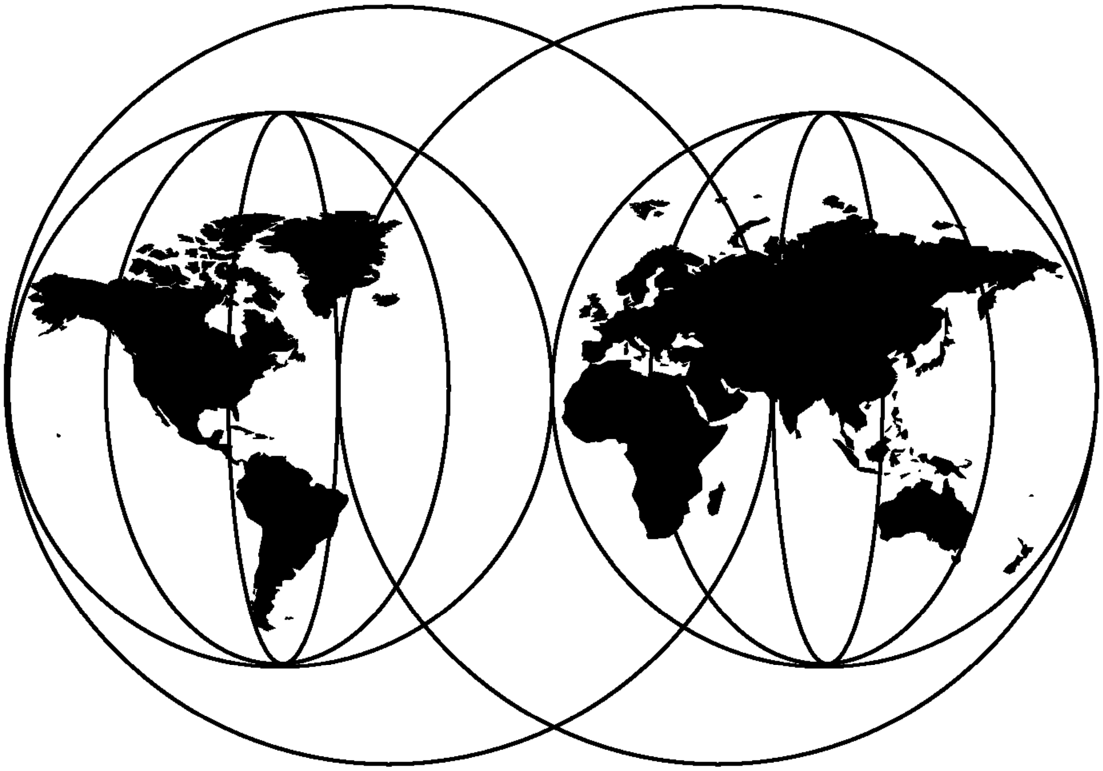
# IMS/ESA V6 Parallel Sysplex Migration Planning Guide for IMS TM and DBCTL

*Bob Gendry, Bill Keene, Rich Lewis, Bill Stillwell, Scott Chen*



**International Technical Support Organization**

http://www.redbooks.ibm.com

SG24-5461-00

International Technical Support Organization

SG24-5461-00

# IMS/ESA V6
# Parallel Sysplex Migration Planning Guide for IMS TM and DBCTL

June 1999

**Take Note!**

Before using this information and the product it supports, be sure to read the general information in Appendix F, "Special Notices" on page 239.

**First Edition (June 1999)**

This edition applies to Version 6, Release Number 1 of IMS/ESA, Program Number 5655-158 for use with the MVS/ESA or OS/390 operating system.

Comments may be addressed to:
IBM Corporation, International Technical Support Organization
Dept. QXXE  Building 80-E2
650 Harry Road
San Jose, California 95120-6099

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

# Contents

**Figures** . . . xi

**Tables** . . . xiii

**Preface** . . . xv
The Team That Wrote This Redbook . . . xv
Comments Welcome . . . xvi

**Chapter 1. Introduction** . . . 1
1.1 Purpose of This Redbook . . . 1
1.2 Organization of This Redbook . . . 1
1.3 Prerequisite Knowledge . . . 2
1.4 Assumptions . . . 2

**Part 1. Developing the Plan** . . . 3

**Chapter 2. Plan Development** . . . 5
2.1 Planning for Migration . . . 5
2.1.1 Planning Phase . . . 5
2.1.2 Preparation Phase . . . 9
2.1.3 Implementation Phase . . . 9

**Part 2. Planning Considerations** . . . 11

**Chapter 3. Planning Considerations with IMS/ESA V6 in Mind** . . . 13
3.1 Discussing the Use of Traditional Queuing and Shared Queues . . . 17

**Chapter 4. Planning Configurations After Failures** . . . 19
4.1 IRLMs . . . 19
4.1.1 Restarting Batch (DLI and DBB) Data-Sharing Jobs . . . 21
4.2 IMS Subsystem Configurations . . . 22
4.3 IMS Data-Sharing Subsystems . . . 22
4.4 IMS Subsystems Utilizing Shared Queues . . . 23
4.5 IMS Subsystems Utilizing VTAM Generic Resources . . . 24
4.6 FDBR Action After an IMS Failure . . . 25
4.7 Restarting BMPs . . . 25
4.8 Degraded Mode Processing . . . 25

**Chapter 5. System Environment Consideration** . . . 27
5.1 Naming Conventions . . . 27
5.2 IMS Subsystem Data Sets . . . 28
5.2.1 IMS Data Sets . . . 28
5.2.2 CQS Data Sets . . . 29
5.2.3 Data Set Characteristics . . . 29
5.3 Executable Code . . . 30
5.3.1 IMS System Code . . . 30
5.3.2 CQS System Code . . . 31
5.3.3 Exit Code . . . 31
5.3.4 Application Program Code . . . 32
5.4 Control Blocks . . . 32

5.4.1 CQS Access . . . . . 32
5.4.2 Online Access . . . . . 32
5.5 Parameter Libraries . . . . . 33
5.6 Dynamic Allocation . . . . . 33
5.7 JCL Libraries . . . . . 34
5.7.1 Procedures . . . . . 34
5.7.2 Jobs . . . . . 34
5.7.3 DBRC JCL Libraries . . . . . 36
5.8 Making the Decision . . . . . 37
5.8.1 Data Sets That Must Be Unique . . . . . 37
5.8.2 Data Sets That Must Be Shared . . . . . 38
5.8.3 Data Sets That Are Probably Shared . . . . . 38
5.8.4 Data Sets That Are Probably Unique . . . . . 40

**Chapter 6. Applications and Databases** . . . . . 41
6.1 Applications . . . . . 41
6.2 Databases . . . . . 42
6.3 Partitioned Applications and Databases . . . . . 42
6.4 Cloned Applications and Data Sets . . . . . 42
6.5 Handling Databases That Are Not Shared . . . . . 42
6.5.1 Routing Transactions . . . . . 42
6.5.2 Copying Databases . . . . . 43

**Part 3. Planning Considerations for IMS TM** . . . . . 45

**Chapter 7. Introduction to IMS TM Considerations** . . . . . 47
7.1 Overview . . . . . 47
7.2 IMS TM Configuration Considerations . . . . . 48
7.2.1 Cloning . . . . . 48
7.2.2 Joining . . . . . 49
7.2.3 Front-End and Back-End . . . . . 50

**Chapter 8. IMS TM Network Considerations** . . . . . 53
8.1 Overview . . . . . 53
8.2 Special Network Considerations . . . . . 54
8.2.1 SLUTYPEP . . . . . 54
8.2.2 ISC . . . . . 55
8.3 APPC (LU 6.2) . . . . . 58
8.4 ETO Considerations with Shared Queues . . . . . 59
8.4.1 Duplicate Terminal Names . . . . . 59
8.4.2 Solutions . . . . . 60
8.4.3 Limiting the Number of End-User Signons . . . . . 61
8.4.4 Dead Letter Queue Considerations . . . . . 61
8.5 Conversational Transaction Processing . . . . . 62
8.6 Application Program Considerations . . . . . 62
8.7 Application Affinities . . . . . 62
8.8 IMS APPLIDs . . . . . 63
8.9 VTAM Model Application Program Definitions . . . . . 64
8.10 A Model Application Program Definition Example . . . . . 64

**Chapter 9. IMS TM Workload Balancing Considerations** . . . . . 65
9.1 Overview. . . . . . 65
9.2 Network Workload Balancing . . . . . 66
9.2.1 Instructions to the End-Users . . . . . 67

9.2.2 USSTAB Processing . . . . . 67
9.2.3 CLSDST PASS VTAM Application Program . . . . . 67
9.2.4 USERVAR Processing . . . . . 68
9.2.5 Use of VTAM Generic Resources . . . . . 69
9.2.6 VTAM Generic Resources for APPC/IMS . . . . . 72
9.2.7 Printers and Network Workload Balancing . . . . . 73
9.2.8 Network Workload Balancing With Session Managers . . . . . 76
9.2.9 Network Workload Balancing With TCP/IP and TN3270 . . . . . 79
9.3 Application Workload Balancing . . . . . 81
9.3.1 Network Workload Balancing . . . . . 81
9.3.2 MSC Balancing . . . . . 82
9.3.3 IMS Workload Router . . . . . 82
9.3.4 Shared Queues and Application Workload Balancing . . . . . 83
9.4 Transaction Scheduling Considerations . . . . . 85
9.5 Summary of Workload Balancing Considerations . . . . . 85

**Chapter 10. IMS TM System Definition Considerations** . . . . . 87
10.1 Master Terminals With IMS/ESA Version 5 . . . . . 87
10.2 Master Terminals With IMS/ESA Version 6 . . . . . 88
10.3 Security . . . . . 89
10.4 Resource Definitions and Usage . . . . . 89
10.5 MSC Considerations . . . . . 90
10.5.1 Inter-IMSplex MSC Without Shared Queues . . . . . 90
10.5.2 Migration to Shared Queues . . . . . 91
10.5.3 Example: Migration of Two MSC-Connected Systems . . . . . 95
10.5.4 Example: MSC Link from a Shared Queues Group Member to a Remote IMS . . . . . 97
10.5.5 Example: MSC Link Between Members of Two Shared Queues Groups . . . . . 99
10.5.6 Example: Multiple MSC Links between Members of Two Shared Queues Groups . . . . . 99
10.5.7 Example: Backup MSC Link Configuration . . . . . 100
10.6 Serial Transactions . . . . . 102
10.6.1 Serial Transactions With Traditional Queuing . . . . . 102
10.6.2 Serial Transactions With Shared Queues . . . . . 102
10.7 Undefined Destinations . . . . . 103
10.7.1 Destination Determination . . . . . 104
10.7.2 Back-End Processing of Input Transactions . . . . . 104
10.7.3 Comments and Recommendations . . . . . 104
10.8 Online Change . . . . . 105
10.9 SPOOL Terminals . . . . . 105

**Part 4. Data-Sharing and Shared Queues Considerations** . . . . . 107

**Chapter 11. Data-Sharing Enablement** . . . . . 109
11.1 IMS System Definition . . . . . 109
11.2 IRLM . . . . . 110
11.2.1 IRLM Subsystem Names . . . . . 110
11.2.2 IRLM as Lock Manager . . . . . 110
11.2.3 IRLM Scope . . . . . 111
11.3 DBRC . . . . . 111
11.3.1 SHARECTL . . . . . 111
11.3.2 SHARELVL . . . . . 111
11.3.3 Skeletal JCL . . . . . 112

11.4 Database Data Set Sharability . . . . . . . . . . . . . . . . . . . . . . . . 112
11.4.1 VSAM SHAREOPTION(3 3) . . . . . . . . . . . . . . . . . . . . . . . . 112
11.4.2 DISP=SHR in JCL or DFSMDA . . . . . . . . . . . . . . . . . . . . . 112
11.5 CFNAMES Statement . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 113
11.6 DEDB Statements in the DFSVSMxx Member . . . . . . . . . . . . . . . 113
11.7 Order of Implementation Steps . . . . . . . . . . . . . . . . . . . . . . . 114
11.8 IMS Procedures for Data-Sharing . . . . . . . . . . . . . . . . . . . . . . 116
11.9 Data-Sharing Performance Considerations . . . . . . . . . . . . . . . . . 118
11.10 Sources of Performance Information . . . . . . . . . . . . . . . . . . . . 120
11.11 Data Sharing Coupling Facility Performance Impacts . . . . . . . . . . . 121
11.11.1 IRLM CF Access . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 122
11.11.2 IMS CF Access . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 123

**Chapter 12. Shared Queues Enablement** . . . . . . . . . . . . . . . . . . . . . 125
12.1 Program Properties Table . . . . . . . . . . . . . . . . . . . . . . . . . . 125
12.2 Structure Definitions . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 125
12.3 Log Stream Definitions . . . . . . . . . . . . . . . . . . . . . . . . . . . . 125
12.4 System Checkpoint Data Sets . . . . . . . . . . . . . . . . . . . . . . . . 125
12.5 Structure Recovery Data Sets . . . . . . . . . . . . . . . . . . . . . . . . 126
12.6 CQS Procedure . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 126
12.7 BPE Configuration PROCLIB Member . . . . . . . . . . . . . . . . . . . 126
12.8 CQS Initialization Parameters PROCLIB Member (CQSIPxxx) . . . . . . 126
12.9 CQS Local Structure Definition PROCLIB Member (CQSSLxxx) . . . . . 126
12.10 CQS Global Structure Definition PROCLIB Member (CQSSGxxx) . . . 126
12.11 Security for CQS Structures . . . . . . . . . . . . . . . . . . . . . . . . 127
12.12 ARM Policy Updates . . . . . . . . . . . . . . . . . . . . . . . . . . . . 127
12.13 IMS Shared Queues PROCLIB Member (DFSSQxxx) . . . . . . . . . . 127
12.14 QMGR and Shared Queues Traces . . . . . . . . . . . . . . . . . . . . 127
12.15 IMS Procedure . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 127
12.16 IMS Procedures for Shared Queues . . . . . . . . . . . . . . . . . . . . 127

**Part 5. MVS Parallel Sysplex Conderations** . . . . . . . . . . . . . . . . . . . . . . . . 129

**Chapter 13. VTAM Generic Resources Enablement** . . . . . . . . . . . . . . 131
13.1 VTAM Requirements . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 131
13.2 IMS Requirements . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 131
13.3 APPC/IMS Requirements . . . . . . . . . . . . . . . . . . . . . . . . . . 131

**Chapter 14. Automatic Restart Manager (ARM)** . . . . . . . . . . . . . . . . 133
14.1 Exceptions to Automated Restarts of IMS . . . . . . . . . . . . . . . . . 133
14.2 Restart Conditions . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 134
14.3 Restart Groups . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 134
14.4 Other ARM Capabilities . . . . . . . . . . . . . . . . . . . . . . . . . . . 135
14.5 ARM with IRLM . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 136
14.5.1 Restarting after IRLM Abends . . . . . . . . . . . . . . . . . . . . . . 136
14.5.2 Restarting after System Failures . . . . . . . . . . . . . . . . . . . . . 136
14.6 ARM with CQS . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 136
14.7 ARM with FDBR . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 137
14.8 Information for ARM Policies . . . . . . . . . . . . . . . . . . . . . . . . 137

**Chapter 15. Coupling Facility** . . . . . . . . . . . . . . . . . . . . . . . . . . 139
15.1 Coupling Facility Planning Guidelines . . . . . . . . . . . . . . . . . . . 139
15.1.1 Structure Placement Rules . . . . . . . . . . . . . . . . . . . . . . . . 139
15.1.2 Initial Structure Placement . . . . . . . . . . . . . . . . . . . . . . . . 140

15.1.3 Structure Sizing . . . . . . . . . . 141
15.1.4 IMS Database Manager . . . . . . . . . . 141
15.1.5 IRLM . . . . . . . . . . 146
15.1.6 Shared Queues . . . . . . . . . . 148
15.1.7 Summary of Structure Characteristics . . . . . . . . . . 150
15.2 Changing CF Structure Sizes . . . . . . . . . . 151
15.2.1 Connection and Structure Persistence . . . . . . . . . . 151
15.2.2 IMS Buffer Invalidation Structure Changes . . . . . . . . . . 151
15.2.3 DEDB VSO Cache Structure Changes . . . . . . . . . . 152
15.2.4 IRLM Lock Structure Changes . . . . . . . . . . 152
15.2.5 Automatic Rebuilds . . . . . . . . . . 153
15.2.6 Shared Queues Structure Changes . . . . . . . . . . 154
15.2.7 Alter and Rebuild for Shared Queues Structures . . . . . . . . . . 154

**Part 6. Operation Considerations** . . . . . . . . . . 157

**Chapter 16. IMS Connections, Security and User Exits** . . . . . . . . . . 159
16.1 IMS Connections . . . . . . . . . . 159
16.2 Security . . . . . . . . . . 160
16.2.1 RACF Security . . . . . . . . . . 160
16.3 IMS SMU Security . . . . . . . . . . 161
16.4 Database Data Set Dispositions . . . . . . . . . . 161
16.5 User Exits . . . . . . . . . . 162
16.5.1 IMS System Exits . . . . . . . . . . 162
16.5.2 IMS Database Manager Exits . . . . . . . . . . 163
16.5.3 IMS Transaction Manager Exits . . . . . . . . . . 163
16.5.4 Common Queue Server Exit Routines . . . . . . . . . . 164

**Chapter 17. IMS and User JCL** . . . . . . . . . . 167
17.1 IMS Procedures . . . . . . . . . . 167
17.1.1 Started Procedures . . . . . . . . . . 167
17.1.2 Executed Procedures . . . . . . . . . . 170
17.2 IMS Jobs . . . . . . . . . . 170
17.3 DBRC Skeletal JCL . . . . . . . . . . 172
17.4 Other Backup and Recovery JCL . . . . . . . . . . 173
17.5 Other Application JCL . . . . . . . . . . 173
17.6 BMP JCL . . . . . . . . . . 173
17.6.1 Using the Function Delivered by APAR PQ21039 . . . . . . . . . . 173
17.6.2 The IMSGROUP Function . . . . . . . . . . 174
17.6.3 Handling IMSID Without APAR PQ21039 Function . . . . . . . . . . 174
17.6.4 Maintenance Levels of RESLIB . . . . . . . . . . 177
17.6.5 Routing BMP Jobs . . . . . . . . . . 177

**Chapter 18. Operations** . . . . . . . . . . 179
18.1 Operational Procedures . . . . . . . . . . 179
18.2 Automated Operations . . . . . . . . . . 179
18.2.1 IMS Time-Control Operations . . . . . . . . . . 179
18.2.2 IMS AO Exits . . . . . . . . . . 180
18.2.3 Other AO Products . . . . . . . . . . 180
18.3 Use of IRLM SCOPE=NODISCON . . . . . . . . . . 180
18.3.1 Use of RDI Regions . . . . . . . . . . 181
18.4 Recovery Procedures . . . . . . . . . . 182
18.5 Support Procedures . . . . . . . . . . 182
18.6 IMS Log Management . . . . . . . . . . 183

18.6.1 IMS Log Archive . . . 183
18.7 Job Scheduling Procedures . . . 183
18.8 Online Change Procedures . . . 184
18.9 IMS Commands from MVS Consoles . . . 184
18.9.1 GLOBAL Commands . . . 185
18.9.2 Alternative to GLOBAL Commands . . . 187
18.9.3 Recommendations for IMS Commands From MVS Consoles . . . 187

**Chapter 19. Fast Database Recovery (FDBR)** . . . 189
19.1 FDBR Monitoring . . . 189
19.1.1 Log Monitoring . . . 189
19.1.2 XCF Status Monitoring . . . 190
19.2 Invoking Recovery . . . 191
19.3 FDBR Failures . . . 192
19.4 Restarting IMS after FDBR Completion . . . 192
19.5 DBRC Authorizations with FDBR . . . 192
19.6 ARM and FDBR . . . 192
19.6.1 ARM Support for IMS With FDBR Active . . . 192
19.6.2 ARM Support for FDBR . . . 193
19.7 FDBR, XRF, and ARM . . . 193
19.7.1 FDBR Advantages and Disadvantages . . . 193
19.7.2 XRF Advantages and Disadvantages . . . 194
19.7.3 ARM Advantages and Disadvantages . . . 194
19.8 Recommendations . . . 194

**Chapter 20. Recovery Procedures** . . . 197
20.1 Image Copies . . . 197
20.1.1 Database Image Copy (DFSUDMP0) . . . 197
20.1.2 Concurrent Image Copy (DFSUDMP0 with CIC Option) . . . 197
20.1.3 Online Database Image Copy (DFSUICP0) . . . 197
20.1.4 Database Image Copy 2 (DFSUDMT0) . . . 198
20.1.5 Summary of Image Copies . . . 198
20.2 Database Recoveries . . . 198
20.2.1 Time-Stamp Recovery Considerations . . . 199
20.3 IMS Batch (DBB and DLI) Job Abends . . . 200
20.4 IMS Online (TM or DBCTL) Abends . . . 200
20.5 IRLM Abends . . . 200
20.6 MVS Failures . . . 200
20.7 Lock Structure Failures . . . 201
20.8 OSAM and VSAM Structure Failures . . . 202
20.9 DEDB VSO Structure Failures . . . 202
20.9.1 Procedure for Failure of One of Two Structures . . . 203
20.9.2 Procedure for Failure of Only Structure or Both Structures . . . 203
20.10 CF Connection Failures . . . 203
20.11 CF Connection Failure to Lock Structure . . . 203
20.12 CF Connection Failure to an OSAM or VSAM Structure . . . 204
20.13 CF Connection Failure to a DEDB VSO Structure . . . 204
20.13.1 Procedure with Connectivity to a Second Structure . . . 205
20.13.2 Procedure without Connectivity to a Second Structure . . . 205
20.14 Disaster Recovery . . . 205
20.14.1 Image Copy Only Disaster Recovery . . . 205
20.14.2 Time-Stamp Recovery Disaster Recovery . . . 206
20.14.3 Latest Archived Log Disaster Recovery . . . 206
20.14.4 Real-Time Electronic Log Vaulting Disaster Recovery . . . 207

**Appendix A. Naming Convention Suggestions** . . . . . . . . . . . . . . . . . . 209

**Appendix B. IMS System Data Sets** . . . . . . . . . . . . . . . . . . . . . . . . 215
B.1 Data Sets That Must Be Unique . . . . . . . . . . . . . . . . . . . . . . . . 215
B.2 Data Sets That Must Be Shared . . . . . . . . . . . . . . . . . . . . . . . . 216
B.3 Data Sets That Are Probably Unique . . . . . . . . . . . . . . . . . . . . . 216
B.4 Data Sets That Are Probably Shared . . . . . . . . . . . . . . . . . . . . . 216

**Appendix C. Sample USERVAR Exit for Network Balancing** . . . . . . . . . . 219

**Appendix D. Parallel Sysplex Publications** . . . . . . . . . . . . . . . . . . . . 231

**Appendix E. Migration Plan Task List** . . . . . . . . . . . . . . . . . . . . . . . 233
E.1 Planning Phase . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 234
E.2 Preparation Phase . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 235
E.3 Implementation Phase . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 237
E.3.1 Data Sharing Environment Implementation . . . . . . . . . . . . . . . 237
E.3.2 Generic Resources Environment Implementation . . . . . . . . . . . 237
E.3.3 Shared Queues Environment Implementation . . . . . . . . . . . . . 238
E.3.4 Implementation of Second System (Clone) . . . . . . . . . . . . . . . 238

**Appendix F. Special Notices** . . . . . . . . . . . . . . . . . . . . . . . . . . . . 239

**Appendix G. Related Publications** . . . . . . . . . . . . . . . . . . . . . . . . . 241
G.1 International Technical Support Organization Publications . . . . . . . . 241
G.2 Redbooks on CD-ROMs . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 241
G.3 Other Publications . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 241

**How to Get ITSO Redbooks** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 243
IBM Redbook Fax Order Form . . . . . . . . . . . . . . . . . . . . . . . . . . . . 244

**Glossary** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 245

**List of Abbreviations** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 249

**Index** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 251

**ITSO Redbook Evaluation** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 253

# Figures

1. Using Unique IRLM Names and Moving IRLMs . . . . . . . . . . . . . . . 20
2. Using Unique IRLM Names and Changing IRLMs Used by IMSs . . . . . 21
3. Using ″IRLM″ for all IRLM Names . . . . . . . . . . . . . . . . . . . . . . . . 21
4. Using One Instance of a Session Manager Without ISTEXCGR . . . . . . 77
5. Using One Instance of a Session Manager With ISTEXCGR . . . . . . . . 77
6. Placing the Session Manager on an MVS Without IMS . . . . . . . . . . . 78
7. Session Managers With Generic Resource Support . . . . . . . . . . . . 78
8. Using DNS With TN3270 and VTAM GR . . . . . . . . . . . . . . . . . . . . 79
9. Using DNS With TN3270 and VTAM GR . . . . . . . . . . . . . . . . . . . . 80
10. Using DNS With TN3270 and VTAM GR After a Failure of IMS1 . . . . . . 81
11. Two IMS systems With an MSC connection . . . . . . . . . . . . . . . . . 95
12. Migration of IMSB to a Shared Queues Group . . . . . . . . . . . . . . . . 96
13. Migration of IMSA and IMSB to a shared queues group is completed . 96
14. MSC link from shared queues group member to remote IMS . . . . . . . 98
15. MSC links between members of two shared queues groups . . . . . . . 99
16. MSC connections from shared queues group members to remote IMS systems . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 100
17. Backing up logical links from a shared queues group to a remote IMS 101
18. Starting a backup link to a shared queues group . . . . . . . . . . . . . 101

# Tables

1. IMS Online Region Library . . . 32
2. IMS Procedures Summary . . . 116
3. IMS Shared Queues Procedure Parameters . . . 128
4. ARM Element Terms . . . 137
5. Structure Characteristics . . . 150
6. Summary of IMS Connectors . . . 159
7. IMSID Usage Before Cloning . . . 174
8. IMSID Usage After Cloning . . . 174
9. Image Copy With Concurrent Updates . . . 198
10. Task List . . . 234
11. Task List . . . 235
12. Task List . . . 237
13. Task List . . . 237
14. Task List . . . 238
15. Task List . . . 238

# Preface

This redbook provides information for those planning for migrating IMS/ESA 6.1 system to a Parallel Sysplex environment using data sharing and shared queues.

The redbook lists all important factors to be consider during the planning stage. It also lists important features and describes the relationships between IMS/ESA V6 and OS/390. Readers will, therefore, be able to understand what benefits can be derived even before the complete implementation to the Parallel Sysplex environment.

The target audience is customer IT architects, system designers, IT managers and IBM representatives who are helping customers on migration projects.

## The Team That Wrote This Redbook

This redbook was written by a team of IMS specialists from the IMS Advanced Technical Support Department at the IBM Dallas Systems Center.

**Bob Gendry** has been a member of the IMS Advanced Technical Support Department at the IBM Dallas Systems Center since 1978 and has over 25 years of experience in providing technical support for IMS. He has given multiple presentations on IMS-related topics to user groups and at the IMS Technical Conferences in the United States and Europe. He has provided assistance to multiple IMS users in planning and implementating IMS in a Parallel Sysplex environment for the past several years. In addition, he has prepared and taught educational materials directly related to the understanding, implementation, and use of IMS in a Parallel Sysplex environment.

**Bill Keene** was a member of the IMS Advanced Technical Support team at the IBM Dallas Systems Center. He has over 25 years of experience in providing technical support for IMS. He is a frequent speaker at GUIDE, SHARE, and IMS Technical Conferences on IMS-related topics. He has provided assistance to multiple IMS users in planning for and implementing the use of IMS in a Parallel Sysplex environment for the past several years. In addition, he has prepared and taught educational materials directly related to the understanding, implementation, and use of IMS in a Parallel Sysplex environment.

After contributing to this redbook, Bill Keene retired from IBM.

**Rich Lewis** has been a member of the IMS Advanced Technical Support Department at the IBM Dallas Systems Center since 1979. He has over 25 years of experience in providing technical support for IMS. Since the introduction of Parallel Sysplex, he has been assisting users in implementing IMS data-sharing. He has written technical documents, created presentations, and developed an IMS Parallel Sysplex data sharing course. He has provided planning services to many customers for the introduction of IMS into their Parallel Sysplex environments. Rich regularly presents Parallel Sysplex topics at IMS Technical Conferences and user group meetings in the United States and Europe.

**Bill Stillwell** has been providing technical support and consulting services to IMS customers as a member of the Dallas Systems Center for 17 years. During that time, he developed expertise in application and database design, IMS performance, fast path, data sharing, planning for IMS Parallel Sysplex exploitation and migration, DBRC, and database control (DBCTL).

He also develops and teaches IBM Education and Training courses, including IMS/ESA Version 6 Product Enhancements, IMS Shared Queues, and IMS Fast Path Implementation, and is a regular speaker at the annual IMS Technical Conferences in the United States and Europe.

**Scott Chen** is a member of International Technical Support Organization, San Jose Center. Scott has installing, configuring, debugging, tuning, consulting, application designing, programming and studying MVS and OS/390 internal, database and transaction management system (includes IMS) and digital library softwares for over 25 years.

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

Dick Hannan
IBM Santa Teresa Laboratory
IMS Development

## Comments Welcome

**Your comments are important to us!**

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

- Fax the evaluation form found in "ITSO Redbook Evaluation" on page 253 to the fax number shown on the form.
- Use the online evaluation form found at `http://www.redbooks.ibm.com/`
- Send your comments in an Internet note to `redbook@us.ibm.com`

# Chapter 1. Introduction

## 1.1 Purpose of This Redbook

This redbook provides information for those creating a plan for migrating an IMS/ESA 6.1 system to a Parallel Sysplex environment using data sharing and shared queues. The reader is assumed to be familiar with the requirements for data sharing and shared queues implementation. This information may be obtained from the IBM Education and Training classes for IMS/ESA Block Level Data Sharing, Shared Queues, and IMS/ESA Version 6 and from *IMS/ESA Data Sharing in a Parallel Sysplex*, SG24-4303.

The redbook applies to both IMS/ESA TM and Database Control (DBCTL) users. Some Parallel Sysplex functions and facilities, such as Shared queues, apply only to IMS Transaction Manager (TM) users. Sections of this redbook which apply only to IMS TM users or only to DBCTL users are identified.

## 1.2 Organization of This Redbook

The main body of this redbook is divided into six sections plus several appendices.

1. Developing the Plan

   This section addresses the plan itself, including the purpose of the plan, its content, and the migration tasks that should be identified within the plan.

2. Planning Considerations

   This section addresses some of the general technical issues that must be considered when developing and implementing the plan and is intended to help make decisions.

3. Planning Considerations for IMS TM

   This section focuses on technical issues related to the IMS TM environment.

4. Data Sharing and Shared Queues Considerations

   This section reviews some technical issues related to IMS data sharing and shared queues.

5. MVS Parallel Sysplex Considerations

   Here, we look at some technical topics related to IMS interact with MVS Parallel Sysplex.

6. Operation Considerations

   Finally, we review technical topics related to IMS operation and recovery procedures.

7. Appendixes

   The appendixes provide additional technical information that might be useful in performing some of the tasks. A sample list of tasks is included at the end of this redbook in Appendix E, “Migration Plan Task List” on page 233. This list includes references to the parts of this redbook that apply to each task in the list.

A list of useful Parallel Sysplex publications, other than the standard IMS/ESA V6.1 product publications, is provided in Appendix D, “Parallel Sysplex Publications” on page 231.

## 1.3 Prerequisite Knowledge

This redbook is written for those who are familiar with the following:

- IMS block-level data sharing definition requirements for IMS, IRLM, and the Coupling Facility
- IMS shared queues definition requirements for IMS (including the Common Queue Server) and the Coupling Facility
- Recovery procedures for failures in the IMS block-level data sharing environment
- Recovery procedures for failures in the shared queues environment
- Roles of IRLM and the Coupling Facility in supporting block-level data sharing
- Roles of the Common Queue Server and the Coupling Facility in supporting shared queues

## 1.4 Assumptions

The assumptions about the installation for which the plan is being developed are:

- The installation is in production with IMS/ESA V6.1 prior to the implementation of this plan.
- A Parallel Sysplex environment has been implemented prior to the implementation of this plan.
- The application (system) to be migrated has been identified.
- The existing IMS environment and its applications are to be **cloned**.

  There is only one current IMS system to be cloned. Its workload will be split across multiple IMS systems which are as identical in function as possible. They can have different workload capacities.

# Part 1. Developing the Plan

# Chapter 2. Plan Development

This section addresses the development of the migration plan and identifies some of the steps and considerations you might encounter when developing the plan. The result of this exercise is not to ″perform″ any of the implementation tasks but to identify those tasks which must be done and to create a plan for accomplishing them. For example, the plan can identify as a task the establishment of a naming convention for system data sets. The naming convention itself is not a part of the plan, but is a result of implementing the plan.

## 2.1 Planning for Migration

The process of migrating to an IMS data sharing and/or shared queues Parallel Sysplex environment should be accomplished in three phases: a planning phase, a preparation phase, and an implementation phase.

### 2.1.1 Planning Phase

The purpose of this phase is to identify/document where you are coming from, where you are going, what you will do if there are failures, and to determine how the plan will be created, who has responsibility, what will be its content and format, what planning or project management tools will be used, and finally to develop the plan itself.

Below, we have identified four major steps in the planning phase. You might recognize fewer or more, but each step below has a purpose, and that purpose, must be satisfied in any planning process.

#### 2.1.1.1 Understand the Existing Environment

The first step in the planning phase is to understand the existing environment. This includes knowing, for each application, the resource requirements (such as, CPU, I/O), service level requirements, schedules and workloads, connections to other subsystems, and the reasons for migrating (for instance, reduced costs or improved performance, capacity, or availability). You should also identify any ″inhibitors″ to migration, such as non-sharable resources.

The assumption here is that the target environment is replacing an existing environment for one or more reasons (for example, capacity, performance, availability, flexibility,...). However, the target environment must continue to provide the equivalent function with performance and availability at least as good as the existing environment. So, in order to define a target environment which will do this, it is first necessary to understand the existing environment. The following describes the characteristics of the existing environment that should be known before defining the target.

- **Why are you migrating to this environment?**

  A major part of developing a migration plan is to choose the configuration to which the migration will be done. This configuration is affected by the reasons for making the migration. The migration to IMS data sharing and shared queues with Parallel Sysplex can be used to provide multiple benefits. These include, but are not limited to:

  - Increased availability

  - Increased capacity
  - Incremental growth capability
  - Operational flexibility

- **What is the current workload?**

  This should be documented in terms that will facilitate the distribution of that workload over two or more (perhaps) processors and should include transaction volumes as well as batch/BMP and support jobs such as image copies, reorganizations, and so forth.

- **Who are the heavy resource users?**

  Which of the above transactions or batch processes require the greatest number of, CPU or I/O resources, and which transactions are the highest volumes. It might be necessary to make special provisions for them in the target environment.

- **What are the service level commitments?**

  What agreements exist for transaction response times, batch elapsed times and availability? Are users billed according to the resources they use?

- **To what systems is the existing IMS connected?**

  This should include other IMSs, DB2s, CICSs, and any other ″intelligent″ systems or devices that might be sensitive to the identity of the IMS system to which they are connected.

- **What are the online and batch schedules?**

  What are the hours of availability for online access and what is the ″batch window″ (if there is one)?

- **Are there any batch or online dependencies?**

  Are there ″sequences″ of processes that must be maintained in the target environment. For example, transactions defined as SERIAL, implying a FIFO processing sequence, should be identified.

- **Are any user exits sensitive to the identity of the IMS system on which they execute?**

  Look particularly at transaction manager exits and system exits as there will be multiple transaction managers with different IDs, connected, perhaps, to different subsystems (for instance, different CICSs or different DB2s) and with only part of the original network.

- **Do any user-written programs process the IMS logs?**

  The logs will be quite different, with each log containing only part of the information that was on the original single-image log.

- **What are the business-critical applications?**

  If one component of the target environment fails (for instance, one of two IMS systems) and can′t be immediately restarted, it might be necessary to quiesce relatively unimportant work on the surviving system in order to shift the entire critical workload to that system. It might also be necessary to shift part (or all) of the network to the surviving system.

- **Are there any non-sharable resources?**

An installation can choose not to share some databases. (See 6.5, “Handling Databases That Are Not Shared” on page 42). These must be identified and plans made for their migration to the target system.

#### 2.1.1.2 Define Target Environment

The next step in the migration planning phase is to define the configuration of the target environment. This includes the number of IMS subsystems in the data sharing group, shared queues group, VTAM generic resource group, the MVS system on which each IMS will run, other subsystems outside of the Parallel Sysplex environment to which the target IMS will connect ( for example, other IMSs, CICS, DB2), the coupling facilities to be used, and which systems will be used for various purposes. For example, one must know on which systems IMS BMPs will be run, or if applications are to be partitioned, on which systems they will run. Be sure the target environment satisfies the reasons for migration.

The elements of the target configuration include the following:

- MVS Systems

  The MVS systems in the target configuration should be identified by the processors and LPARs on which they run and the types of work they will support. The types of work include the IMS subsystems and support processes they will handle.

- IMS Subsystems and Processes

  These are IMS online systems (either IMS Transaction Manager, DCCTL, or DBCTL), IMS batch jobs, and IMS support processes. Support processes include database image copies, database reorganizations, log archiving, and definition jobs. The MVS systems on which they will run should be identified.

  The use of each IMS online system should be identified.

  – IMS TM

    For IMS TM this includes the terminal networks that will use them, the ISC and MSC connections to other systems, APPC connections, the associated DB2 subsystems, the application programs, and transactions that will run on each system. Application programs include BMPs.

    For IMS TM and DCCTL subsystems, special planning considerations will be required if shared queues, VTAM generic resources, and/or other Parallel Sysplex enhancements delivered with IMS/ESA V6.1 are to be used.

  – DBCTL

    For DBCTL this includes the CICS systems that will attach to them, the DB2 systems used by BMPs, and the application programs that will run on each system. Application programs include BMPs.

  For cloned systems, it is assumed that all online transactions and programs will be run on each system. For performance, operational, or affinity reasons, there may be exceptions. These should be understood, and the target configuration must account for these considerations. Typically, BMPs and IMS batch jobs will be directed to particular IMS or MVS systems. Many installations will want to run them on only one MVS system.

- Coupling Facilities

The coupling facilities and coupling facility structures to support IMS should be identified. This includes structure sizes and the placement of these structures in support of:

- Data sharing structures
  - IRLM (lock structure)
  - VSAM buffer invalidation structure (directory-only cache structure)
  - OSAM buffer invalidation/cache structure (store-through cache structure)
  - Shared DEDB VSO structure(s) (store-in cache structure)
- Shared Queues
  - Message queue and EMH queue primary and overflow structures (list structures)
  - MVS logger structures (list structures)
- VTAM Generic Resource structure (list structure)

### 2.1.1.3 Define Degraded Mode Environment

Next you must decide what you will do if something fails. Since a Parallel Sysplex consists of multiple MVS and IMS systems, an installation should plan what it will do if one or more components fail. For example, there may be certain applications or systems that are more critical to the business and therefore should have preference to be available when there is an outage of part of the system. This is called **degraded mode** processing.

During this phase, you should determine both the processing and business impact of the failure of any component of the target environment. Identify those applications which must be given priority in a degraded processing environment. You must also consider what users who are connected to a failed component should do (such as, log on to another IMS?).

Some tasks which should be included in this phase are:

- Perform a ″component failure impact analysis (CFIA)″ for critical components
- Prioritize applications for degraded mode processing
- Identify applications to run in degraded mode
- Identify network terminals and connections to be reconnected to another system if one system fails

### 2.1.1.4 Develop the Plan

The plan should recognize the following two phases of the migration process: preparation and implementation. Although this document does not prescribe a format for the migration plan, the following elements should be included:

- Tasks - What must be done?
- Responsibility - Who is responsible to see that it gets done?
- Dependencies - Is any task a prerequisite to another task?
- Duration - How long should each task take?
- Schedule - When must it be done - start/complete/drop-dead dates?
- Status - A mechanism for monitoring progress?

Appendix E, “Migration Plan Task List” on page 233 is a list of tasks that have been identified which should be a part of the migration plan.

### 2.1.2 Preparation Phase

Most of the tasks identified in the migration plan are implemented during the preparation phase. The plan may say, for example, that a naming convention must be established for system data sets. During this phase, that naming convention will be developed. Or the plan may say that operational procedures must be updated. During this phase, those procedures are updated.

Some of the tasks in this phase will be ″decision″ type tasks (for instance, how many copies of RESLIB do I want?). Others will be ″implementing″ some of these decisions (such as, making two copies of RESLIB). At the conclusion of this phase, you are ready to migrate your existing system to production.

### 2.1.3 Implementation Phase

The final phase in the migration process is the actual implementation. That is, the existing environment will be converted to an operational IMS Parallel Sysplex environment. The actual ″implementation plan″ will probably be produced as part of the preparation phase as it is unlikely that enough information will be available during planning to generate this plan.

# Part 2. Planning Considerations

# Chapter 3. Planning Considerations with IMS/ESA V6 in Mind

IMS/ESA V5.1 was the first release of IMS to exploit Parallel Sysplex facilities. It supported data sharing for full function and Fast Path databases. IMS/ESA V6.1 contains many enhancements for the use of IMS in a Parallel Sysplex. These enhancements and others are listed below with brief comments about their use or impact on IMS systems. Wherever appropriate, planning and migration considerations for these enhancements are explained in the following sections of this publication.

- **Shared Queues** is a replacement for the manner in which input transactions and output messages are physically managed and queued for processing. With Shared Queues, the use of the message queue data sets has been eliminated and replaced by queuing messages to list structures in one or more coupling facilities.

  Since a Coupling Facility can be accessed by multiple IMS subsystems, the messages queued in a Coupling Facility can be accessed and processed by any IMS that is interested in processing a particular resource name. For example, all IMSs with access to the queue structures and with TRANA defined are ′interested′ in processing TRANA. A given instance of TRANA, therefore, can be processed by any IMS with access to the shared queues on the Coupling Facility regardless of which IMS queued TRANA to the shared queues as a result of terminal input or application output. This example of ′sharing′ the processing load among multiple IMS subsystems is one of the key benefits of shared queues: That is, shared queues enable:

  - Workload balancing. Multiple IMS subsystems can balance dependent region application workloads among themselves automatically.
  - Improved availability. If one IMS fails, the surviving IMSs can continue processing.
  - Incremental growth. IMS subsystems can be added to the Parallel Sysplex environment without disruption.

- **VTAM Generic Resource** support allows terminals to log on using a generic name rather than a specific IMS APPLID when requesting a session with one of the IMSs in a Generic Resource Group. VTAM Generic Resource support selects one of the IMS subsystems in the generic resource group to which the session request will be routed. VTAM′s session selection algorithm enables the session requests to be evenly distributed across the IMS subsystems to achieve network workload balancing while simplifying the terminal end-user interface to the IMSs in the Parallel Sysplex.

- **Data sharing for DEDBs with SDEPs and for DEDBs with the VSO option** is enabled with IMS/ESA V6.1. The lack of data sharing support for these particular types of DEDBs were considerations when migrating to data sharing in a Parallel Sysplex environment with IMS/ESA V5.1.

- **OSAM Coupling Facility Caching** allows OSAM blocks to be cached in a coupling facility structure.

  One of the costs of data sharing among IMS subsystems is known as ′buffer invalidation.′ Whenever a data sharing IMS updates a block in its local buffer pool, if the block is also in another system′s buffer, that buffer must be invalidated. A future reference by that IMS to the block in the invalidated buffer requires it to reread the block from the database data set on DASD.

With OSAM Caching capability, IMS users now have the option of storing updated blocks in a coupling facility structure such that references to invalidated blocks can be satisfied from the coupling facility, thus avoiding a reread from a data set on DASD.

The OSAM Caching capability has three options: Store all selected database data set blocks in the coupling facility or, store only blocks that have been changed in the coupling facility, or, do not store any OSAM blocks in the coupling facility.

- **Fast Database Recovery** (FDBR) is a facility or function that addresses a data-sharing problem known as the ″retained locks problem.″

  At the time a data sharing subsystem fails, it might hold non-sharable locks on database resources that cannot be released until the retained locks are released. Retained locks are usually released by dynamic backout during emergency restart of the failed IMS. The execution of a successful emergency restart typically takes a few minutes, but can take much longer depending upon the type of failure and nature of the emergency restart.

  Given that a data-sharing IMS has failed, the other data sharing partners are still executing. If application programs executing on the the remaining IMS subsystems try to access a database resource that is protected by a retained lock, the program is abended with a U3303 abend code, and the input transaction is placed on the suspend queue. If access attempts to retained locks result in many U3303 abends, then the impact on the application programs executing on the surviving IMSs can be severe (that is, application programs are unable to get their work done, and terminal operators might be hung in response mode).

  Application program failures because of retained locks can be avoided or minimized in two ways. Application programs can be changed to issue an INIT STATUS GROUPA call and to check for BA and BB status codes for calls to databases. These application changes eliminate the U3303 abend when a database call attempts to access a resource protected by a retained lock. This method of avoiding application abends because of access to retained locks has been available with IMS for many years. Unfortunately, not many IMS users have modified their application programs to take advantage of checking for the BA and BB status codes.

  FDBR provides a second way to address the retained lock problem by dynamically backing out in-flight units of work whenever an IMS subsystem in a data-sharing group fails while holding retained locks. FDBR minimizes the impact of the problem by reducing the amount of time that retained locks are held.

- **Sysplex communications** for and among the IMSs in a Parallel Sysplex environment has been enhanced by:

  – Allowing commands to be submitted to any IMS TM, DCCTL, or DBCTL subsystem in a Parallel Sysplex environment from a single MCS or E-MCS console in the Sysplex.

  – Routing synchronous and asynchronous responses to a command to the MCS or E-MCS console which entered the command.

  – Providing RACF (or equivalent product) and/or Command Authorization Exit (DFSCCMD0) security for commands entered from MCS or E-MCS consoles.

These improvements in Sysplex communications capability allow users to control multiple IMSs within a Parallel Sysplex from a single MVS console rather than from multiple MVS consoles or IMS Master Terminals.

- **MSC dynamic MSNAMEs and shared local SYSIDs** are functions designed explicitly to assist in the maintenance of IMS systems in a shared queues group when one or more of the IMSs in the group has external MSC links to other IMS systems outside the group or when such MSC links are added to one or more of the IMSs in the group. As the label implies, ′dynamic MSNAMEs′ creates MSNAMEs dynamically whenever an IMS joins the group and does not have the MSNAMEs defined to the existing members of the group defined to it or the existing members of the group do not have the new shared queues group when member′s MSC definitions defined to them.

  Dynamic MSNAMEs, as implemented, remove the requirement for each IMS in a shared queues group to SYSGEN the definitions of the MSC network. In some situations, an IMS remote to the shared queues group might be added to the MSC network with minimal disruption to the IMSs in the shared queues group.

  Shared local SYSIDs among the members of a shared queues group allow all members of the group to properly handle MSC-related messages that are received or sent by members of the group from/to IMS systems connected through IMS to one or more members of the group.

- **Primary Master/Secondary Master Terminal** definitions and use:

  One of the advantages of a Parallel Sysplex environment is the capability to clone the IMS system definitions for all of the IMSs, thus making the definition process less error prone. With IMS/ESA V5.1, the Master and Secondary Master Terminal Operator physical terminal definitions could be cloned if these physical terminals did not actually exist. Some automation products have the capability of providing the Master Terminal function. Also, the Master and Secondary Master LTERM definitions can be cloned because the LTERMs are directly associated with the IMS system that had output queued to them.

  Shared queues with IMS/ESA V6.1 introduces the possibility of duplicate LTERM names. With shared queues, duplicate LTERM names associated with active sessions are to be avoided because the delivery of output messages then becomes similar to a lottery for the delivery of those output messages. The results of the lottery (competition to deliver Master Terminal output) would sometimes deliver an output message to the correct Master Terminal and sometimes not. ′Sometimes not′ is unacceptable, and without a change to IMS, would require the system definitions of all of the IMSs in a shared queues group to define unique Master and Secondary Master Terminal LTERM names.

  IMS/ESA V6.1 allows the physical terminal and LTERM name definitions for the Primary and Secondary Master Terminals defined in an IMS system definition to be overridden at execution time through a new, unique proclib member associated with each IMS subsystem. This new proclib member eliminates the problem previously described.

IMS/ESA V6.1 also includes new, or enhancements to existing, functions that enhance the use of IMS in a Parallel Sysplex environment. These enhancements are also available to IMS subsystems outside the Parallel Sysplex arena. A list of these enhancements follows:

- **The** /STOP REGION ABDUMP **command** has been enhanced to detect whether the region which is to be stopped is waiting on a lock request within the IRLM. If it is waiting on a lock, when the command is issued, the waiting dependent region task is resumed and forced to return to its dependent region address space, where it is abended (U0474).

  In prior releases, the execution of the command was waited upon if the dependent region to be aborted was waiting for an IRLM lock. If the wait for the command to execute is excessive, the operator wanting to terminate the dependent region can issue another form of the command, /STOP REGION CANCEL, to force the region to terminate. This form of the command causes the entire IMS online system to abnormally terminate if the dependent region were waiting on a lock.

- **CI reclaim** is not turned off in data-sharing or XRF environments with IMS/ESA V6.1. It was turned off in prior releases, which had the potential for causing performance problems when using primary and secondary indexes.

- **Improved security** is accomplished with IMS/ESA V6.1 by propagating the security environment to a back-end IMS when it is processing a transaction that requires signon security. This is an important enhancement in either a shared queues or MSC environment.

- **UCB VSCR/10,000 DD-names** provide virtual storage constraint relief for private address space storage below the 16 MB line and the ability to allocate as many as 10,000 data sets to an address space. Note, there is a limit of 8,189 full-function data base data sets that can be opened concurrently. For DEDBs, which are allocated to the control region, 10,000 area data sets can be opened as well as allocated. These enhancements allow the number of data sets that can be allocated and opened in the Control Region and DL/I SAS address spaces to be greatly increased.

- **DBRC performance** has been enhanced in several ways. These performance enhancements are an integral part of IMS/ESA V6.1. Nothing has to be done by the IMS user to receive the benefit.

- **Increased number of MSC links and SYSIDs** gives the ability to define more MSC resources to an MSC network. These changes remove the concern of the addition of multiple IMS TM systems to an MSC network in a Parallel Sysplex environment.

- **Generic /START region** capability allows a single proclib member to be used to start a dependent region under any IMS TM system. This is useful when starting Message Processing Regions (MPRs) and Interactive Fast Path Regions (IFPs) and can have some use when starting Batch Message Processing (BMP) Regions.

- **Online change for DEDBs** allows changes to be made without bringing down IMS TM.

- **SMS Concurrent Copy** is a new form of image copy that produces an image copy with high performance with no or minimal data base data set outage. The fuzzy image copy allows the data base data set to be copied to remain fully accessible by the online system(s). A clean image copy does require a short outage (a few seconds). SMS Concurrent Copy supports all types of database data set organizations (KSDS, ESDS, and OSAM) used by IMS.

- **Conversational SPA Truncated Data Option** supports a variable length SPA size when conversational processing involves multiple program-to-program switches. This support is useful when the processing IMS is not the owner of the conversation, such as in a shared queues environment.

- **DBRC NOPFA Option** prevents the setting of the *Prohibit Further Authorization* flag when the GLOBAL form of a database command is issued, such as /DBR, /STOP, or /DBD with the GLOBAL keyword option. If the intent of issuing the command, for example, /DBR DB GLOBAL, was to deallocate a database from multiple data sharing IMS TM systems so that stand-alone batch processing could be started against the database, the use of the NOPFA keyword option with the /DBR command eliminates the need to turn off the ′Prohibit Further Authorization′ flag in DBRC before starting the stand-alone batch processing.
- **DBRC DST/Year 2000:** DBRC DST support eliminates the need to shut down IMS twice a year when the switch to or from daylight savings time (DST) takes place. The elimination of the requirement to shut down twice a year is only true when all of the IMS systems sharing the RECON data sets are at the IMS/ESA 6.1 level.

  DBRC Year 2000 support provides support for a four-digit year not only internally within IMS but also externally for IMS users, such as, application programs executing in a dependent region.
- **BMP Scheduling Flexibility:** APAR PQ21039 for IMS/ESA V6 simplifies the specification of the IMSID for users whose BMPs can execute under multiple control regions in a Parallel Sysplex. Without this enhancement, moving a BMP from one control region to another typically requires a change in the BMP IMSID parameter. With this enhancement, no changes to BMP JCL are required even when a BMP is executed under different control regions. A detailed description of this enhancement is found in 17.6.1, “Using the Function Delivered by APAR PQ21039” on page 173.

## 3.1 Discussing the Use of Traditional Queuing and Shared Queues

The use of shared queues is optional. With IMS/ESA V6.1, one can choose to use the new shared queues function or to use traditional queuing (for example, continue to use the message queue data sets). Depending upon which form of queuing is to be used, the Parallel Sysplex implementation considerations can be quite different. For this reason, the discussions that follow differentiate between the implementation of IMS in a Parallel Sysplex environment using traditional queuing versus shared queues.

# Chapter 4. Planning Configurations After Failures

An installation should have a plan for what it will do when any component of the Parallel Sysplex fails. For example, if a processor is unavailable, the installation should have a plan for what subsystems, such as IMSs or IRLMs, will be moved to another processor. Decisions about the configurations that will be used after failures will determine what will be included in recovery procedures. These decisions can also affect what names are chosen for subsystems. This means that these decisions must be made early in the planning process.

Components that might fail include a processor, an MVS system, an IRLM, an IMS subsystem, a CQS address space, an FDBR address space, a component of the MVS logger, one or more Coupling Facility links, a Coupling Facility structure, or a Coupling Facility. Some failures can be addressed by moving a component. This, in turn, might affect the execution of other components. For example, the movement of an IMS subsystem to a different MVS might affect where BMPs are run.

The following are considerations that should be understood in making decisions about configurations after failures.

## 4.1 IRLMs

The following rules apply to the use of IRLMs in a data-sharing group.

- Each IMS subsystem must specify the name of the IRLM that it uses. This is the IRLMNM parameter.
- An IMS can be restarted using any IRLM in the data-sharing group. This applies to both normal and emergency restarts.
- IRLM names are subsystem names and all subsystems on an MVS must have unique names.
- IRLM names do not have to be unique in a data-sharing group. IRLMs running on different MVS systems can use the same IRLM name.
- IRLM IDs must be unique in a data-sharing group. IRLMs must have different IDs even when they run on different MVS systems.
- IRLM names of all IRLMs that will be run on an MVS must be known to that MVS.

An installation can choose to have either a unique name for each IRLM in the data-sharing group or a common name shared by all of the IRLMs. If a common name is used, an IMS can be restarted on any MVS without changing the IRLMNM parameter for the IMS. Of course, only one IRLM per MVS need be used for the data-sharing group. If unique names are used, an installation has two options for restarting an IMS on a different MVS. It can either change the IRLM name used by the IMS or start the second IRLM on the MVS before restarting IMS.

An IMS batch (DLI or DBB) data-sharing job specifies the IRLM name. If movement of work requires that the batch job use an IRLM with a different name, its JCL must be changed. If an installation wants to be able to run an IMS batch job on any MVS system without changing its JCL, all IRLMs must have the same name.

The IRLM name is made known to MVS by specifying it in the IEFSSNxx member of the MVS PARMLIB. MVS is preconditioned with the names of IRLM and JRLM; so these names do not have to be included in IEFSSNxx. All other IRLM names that might be used on an MVS must be specified in its IEFSSNxx member.

The following figures illustrate three cases for handling the failure of an MVS system.

- Figure 1 shows the use of unique IRLM names an moving an IRLM and its associated IMS when their MVS fails.
- Figure 2 on page 21 shows the use of unique names for all of the IRLMs in the data sharing group and moving only the IMS when an MVS fails. In this case, the IRLMNM parameter for the IMS must be changed to match the IRLM on the MVS to which it is moved.
- Figure 3 on page 21 shows the use of a common name by all IRLMs. When an MVS fails, only its IMS is moved to another MVS. The IRLM does not have to be moved and the IRLMNM specified by the moved, IMS does not have to change.

All figures show the movement of IMSA from MVSA to MVSB after MVSA becomes unavailable.

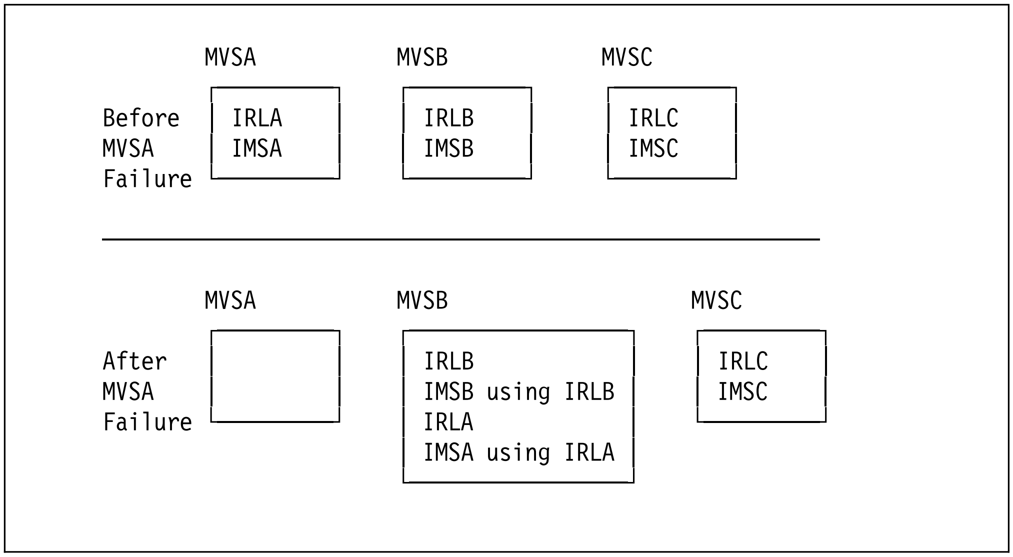


*Figure 1. Using Unique IRLM Names and Moving IRLMs*

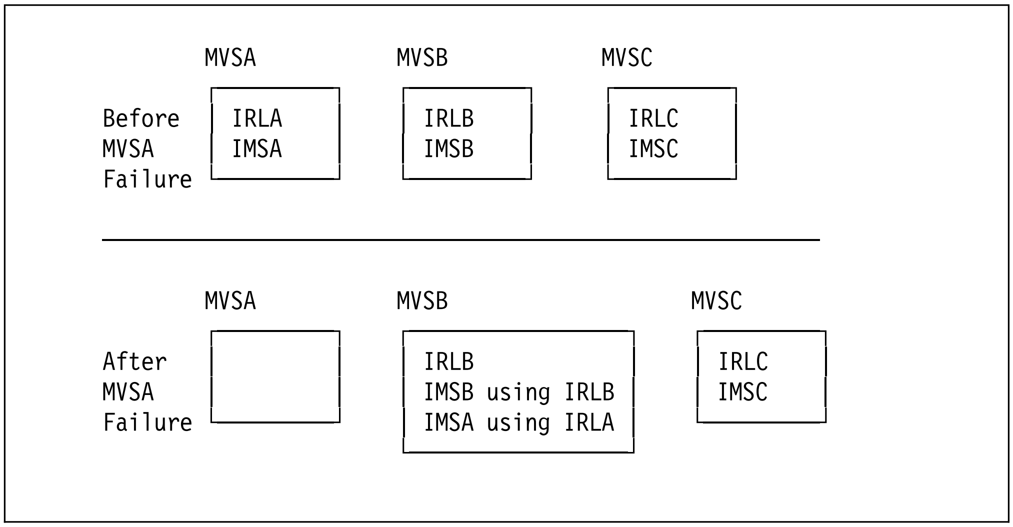


*Figure 2. Using Unique IRLM Names and Changing IRLMs Used by IMSs*

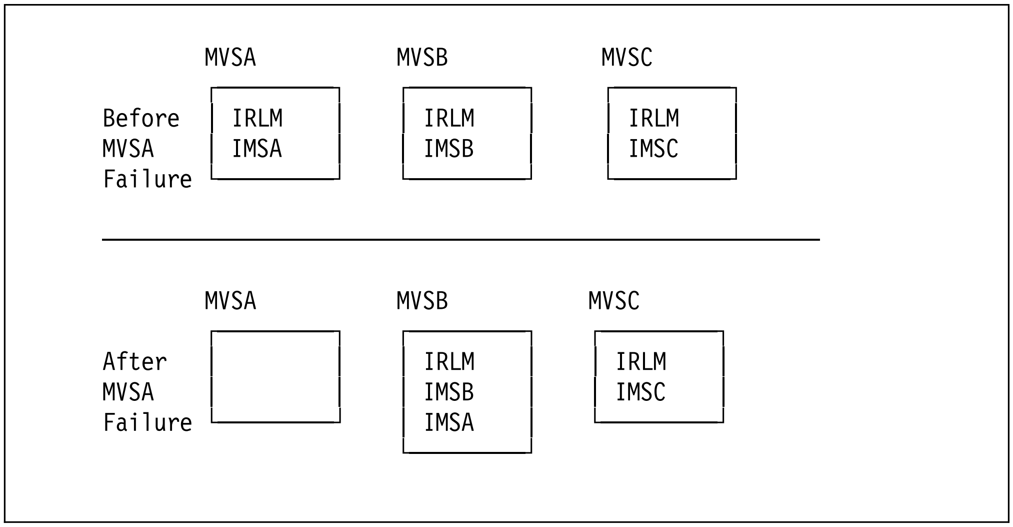


*Figure 3. Using ″IRLM″ for all IRLM Names*

## 4.1.1 Restarting Batch (DLI and DBB) Data-Sharing Jobs

When a data-sharing batch (DLI or DBB) job fails, dynamic backout generally is not invoked. The only exception is for IMS pseudo abends of jobs using a DASD log and specifying BKO=Y. In general, batch backout must be run for an updater. In a data-sharing environment, this backout can be run using any IRLM in the data-sharing group. Of course, the batch log must be available to the backout. It is usually important to run this backout as quickly as possible since any modify locks held by the failed batch job are retained and cause requestors of the locks to receive a ″lock reject″ condition. Lock rejects usually result in U3303 abends.

After a batch job is backed out, it will normally need to be restarted. The restart of the batch job can be done using any IRLM in the data-sharing group. If the restarted batch job reads a checkpoint record from its log, this log must be available on the MVS system used for restarting the batch job.

## 4.2 IMS Subsystem Configurations

The exploitation of a Parallel Sysplex environment by IMS began with IMS/ESA V5.1 and the changes introduced in that release to enhance data-sharing in a Parallel Sysplex environment. IMS subsystems in a Parallel Sysplex environment that are sharing the same databases are known as a data- sharing group. It is not required that all of the IMS subsystems in the Sysplex be in a data-sharing group, or be in the same data-sharing group. For example, some of the IMSs in the Sysplex could have been configured as TM front-ends to transaction processing back-end IMSs that are connected to the front-end IMSs using multiple systems coupling. It is also valid for these back-end IMSs to be configured into multiple data-sharing groups.

IMS/ESA V6.1 introduces three new types of groups. These are: Shared queue, VTAM generic resource, and FDBR groups.

A shared queues group consists of those IMS TM systems that are sharing the same message queues. There can be multiple shared queues groups within a Parallel Sysplex.

Similarly, a VTAM Generic Resource Group consists of those IMS TM systems that have joined the same group. There can be multiple VTAM Generic Resource Groups configured within a single Parallel Sysplex environment.

A given IMS and its associated FDBR both join a unique group. For example, IMSA and FDBRA join the same group, IMSB and FDRB join another unique group, and so on.

Some general statements can be made about these groups:

- A given IMS subsystem can be a member of only one group of each type. That is, a given IMS can be, at most, a member of one data-sharing group, one shared queues group, one VTAM Generic Resource Group, and one FDBR group.
- The members of a group must be identified uniquely within a particular group. Group members are identified by their IMSIDs, which must be unique. The use of FDBR is an exception to this statement. An IMS and its associated FDBR join a group whose name is, by default, FDR + IMSID.
- Multiple groups of a given type, for example multiple data-sharing groups, can coexist within the same Parallel Sysplex environment.
- There are multiple IMS subsystem types: DB/DC, DCCTL, DBCTL, stand-alone batch, and some IMS utilities. Only DB/DC and DCCTL subsystems can be members of shared queues and VTAM Generic Resource groups. With the exception of DCCTL subsystems all subsystem types listed above can be members of a data sharing group.

## 4.3 IMS Data-Sharing Subsystems

The following rules apply to the use of IMS subsystems in a data-sharing group.

- Each IMS subsystem in the data-sharing group must have a unique subsystem name.
  - The subsystem name for an IMS TM or DBCTL system is the IMSID.
  - The subsystem name for an IMS batch (DBB or DLI) job is the jobname.

- Without the enhancement delivered with APAR PQ20139, a BMP must specify (or default to) the subsystem name of the control region under which it is to execute. See 17.6.1, “Using the Function Delivered by APAR PQ21039” on page 173.

  The default control region is determined by the DFSVC000 module in the STEPLIB concatenation. The IMSID parameter on the IMS system definition IMSCTRL macro determines the default in DFSVC000.

- A CICS system must specify the subsystem name for the control region it will use.

  This is specified by the DBCTLID in the DRA startup parameter table.

An installation can choose to move a BMP from one IMS subsystem to another because an IMS subsystem is unavailable or because the workload on a system exceeds its capacity. In any case, if a BMP is moved to another IMS subsystem, its IMSID must be changed. Alternatively, an IMS subsystem can be moved from one MVS to another to allow its BMPs to be moved without changing their IMSID specifications.

Similarly, a CICS system can be moved from one IMS subsystem to another. If this is done, the DBCTLID specification in the DRA startup table must be changed or overridden. Of course, an IMS subsystem can be moved from one MVS to another to allow its CICSs to be moved without changing their DBCTLID specifications.

## 4.4 IMS Subsystems Utilizing Shared Queues

DB/DC and DCCTL are the two types of IMS subsystems that can utilize shared queues. One can choose to use shared queues or to use traditional queuing, not both. It is expected that the migration to IMS V6 will be in two steps with regard to message queuing. First, IMS V6 will be brought into production using traditional queuing. Once these systems have proven themselves— that is, fallback to a prior release has been abandoned—the conversion of these systems to utilize shared queues can begin.

The following rules apply to the use of those subsystems that are in the same shared queues group:

- Each IMS subsystem in the shared queues group must have a unique subsystem name.
  - The subsystem name for a DB/DC or DCCTL system is the IMSID.
  - To utilize shared queues, each IMS that is to be a member of the same shared queues group must specify the same shared queues group name,SQGROUP=, in the DFSSQxxx proclib member that is specified on the control region execution procedure (SHAREDQ=xxx, where xxx is the suffix of the DFSSQxxx proclib member).
- Each IMS in a shared queues group is associated with a Common QUEUE Server (CQS) address space. Each CQS in the shared queues group has a unique name. Note, each CQS name is specified in a proclib member, CQSIPxxx, unique to each CQS. Although the keyword used to specify the name, SSN=, might imply that each CQS is an MVS subsystem, this implication is incorrect. CQS does not execute as an MVS subsystem.

  - When IMS initializes and joins a shared queues group, it either reconnects to its active CQS or starts its associated CQS address space if it is not active.
  - IMS and CQS are failure independent. When one fails, the other stays active.
  - When IMS fails, it is only necessary to restart the failed IMS on the same MVS. Nothing need be done with CQS. IMS will automatically reconnect to or, if necessary, start its associated CQS.
  - When CQS fails, it need only be restarted using the MVS START command or as an MVS batch job. CQS can optionally be restarted using ARM.
  - Users of shared queues have the option of (a) terminating or (b) not terminating CQS when IMS is normally shut down. When IMS is restarted, IMS will attempt to resynchronize with its partner CQS. CQS uses its last recorded system checkpoint to determine where to begin reading the MVS log forward to accomplish the resynchronization. If CQS is not terminated when IMS is shut down, CQS does not have to read the MVS log when IMS is restarted.

## 4.5 IMS Subsystems Utilizing VTAM Generic Resources

- To utilize VTAM generic resources, each IMS that is to be a member of the same generic resource group must specify the same generic resource group name, GRSNAME=, in the IMS Proclib member, DFSPBxxx.
- Each IMS within a generic resource group must have a unique APPLID.
- If an IMS subsystem using VTAM generic resources fails, terminal users simply log on again to the same VTAM generic resource name, which will result in establishing a new session with one of the surviving IMSs in the generic resource group.
- Depending upon the generic resource execution option selected, the reestablishment of a session after an MVS failure is affected as follows:
  - If IMS is selected to manage affinities and an IMS subsystem fails as a result of an MVS failure, affected terminal users must wait until the failed IMS is restarted on any MVS within the Parallel Sysplex before their sessions can be reestablished.
  - If VTAM is selected to manage affinities and an IMS subsystem fails as a result of an MVS failure, affected terminal users can immediately reestablish a session with one of the surviving members of the generic resource group by simply logging back on using the generic resource name.
- VTAM Generic Resource support for APPC/IMS is provided by APPC/MVS. It uses a different generic resource group from that used by IMS. The APPC/IMS group is specified by the GRNAME= keyword on the LUADD PARMLIB statement for each APPC/IMS instance.

## 4.6 FDBR Action After an IMS Failure

FDBR monitors an IMS subsystem using the IMS log and XCF status monitoring services. When it determines that IMS has failed, FDBR dynamically recovers databases by backing out uncommitted updates for full-function databases and executing redo processing for DEDBs. These are the functions that would be performed by emergency restart.

When FDBR completes the back outs and redoes, it releases the locks held by the failed subsystem. This allows any sharing subsystems to acquire these locks. From the time of the IMS failure until the locks are released, other subsystems requesting the locks receive lock rejects. Lock rejects usually result in U3303 abends of application programs on the other subsystems.

FDBR optionally can be automatically restarted by ARM.

For further details on FDBR, read Chapter 19, "Fast Database Recovery (FDBR)" on page 189.

## 4.7 Restarting BMPs

When a BMP fails, dynamic backout is invoked by its IMS subsystem. For BMP abends, this is done immediately. If the IMS subsystem fails at the same time, the backout occurs as part of emergency restart processing for the IMS subsystem. After a BMP is backed out, it will normally need to be restarted. It might be possible to restart the BMP on another IMS, or it might be necessary to use the IMS on which the BMP was running when it failed. This depends on the design of the BMP and the installation′s operational procedures.

- If the BMP restart is done by specifying CKPTID=′LAST′, the same IMS must be used. In this case, the IMS system determines the last checkpoint ID.
- If the BMP restart specifies a CKPTID and the checkpoint records are read from the OLDS, the same IMS must be used.
- If the BMP restart specifies a CKPTID and the checkpoint records are read from a data set identified in a //IMSLOGR DD statement, any IMS subsystem in the data-sharing group can be used.
- If the BMP restart does not specify a CKPTID and IMS restart facilities are not used (the BMP has its own application logic for restart), any IMS subsystem in the data-sharing group can be used.

## 4.8 Degraded Mode Processing

When one component of the Parallel Sysplex is not available and cannot be restarted quickly, the remaining components can not have the capacity to provide adequate service for all applications. In this case, the sysplex enters a degraded mode of processing and some applications can have to be stopped to allow higher priority applications to continue providing good service. When this occurs, procedures must be in place to move the critical workload to the surviving system(s) and to stop that workload which is less critical, or at least less time sensitive. To achieve this workload movement and any other adjustments, it is necessary to generate a prioritized list of those applications that can be stopped.

# Chapter 5. System Environment Consideration

This chapter outlines a number of environmental elements that should be considered.

## 5.1 Naming Conventions

Each IMS system will require many of the same components. Most of these components have names by which they are identified. To avoid confusion, a naming convention which addresses the issue of multiple, similar systems should be developed. The following identifies those components for which a naming convention is recommended:

- MVS names
- IMS data-sharing group names
- IMS shared-queue group names
- IMS VTAM generic resource group names
- Subsystem names
  - IMS
  - FDBR
  - IRLM
  - DB2
  - CICS
- Other address spaces
  - CQS
- Data set names
  - IMS system data sets
  - CQS checkpoint data sets
  - Structure recovery data sets
  - MVS logger data sets
  - Database data sets
  - Application data sets
- Job names for
  - IMS regions
  - IRLM regions
  - CQS regions
  - FDBR regions
- Started task names for
  - IMS regions
  - IRLM regions
  - CQS regions

  - FDBR regions
- Coupling Facility structure names

Of course, many installations already have names and naming conventions and will probably want to modify their current names and conventions as little as possible. Appendix A, "Naming Convention Suggestions" on page 209 addresses some of the requirements and makes suggestions for names.

## 5.2 IMS Subsystem Data Sets

This discussion of data sets assumes that the IMSs in the Parallel Sysplex are actively involved in the use of data sharing and shared queues. Because there are multiple subsystems (IMSs) and address spaces (CQSs) involved and because each requires access to the same or similar data sets, some decisions must be made as to how they are to be implemented in the parallel environment. These decisions are generally limited to the following options:

- **Must be unique**

  Each subsystem or address space must have its own (unique) version of this data set. It is not shared with any other subsystem. These are data sets that are updated by IMS or CQS.

- **Must be shared**

  Each subsystem or address space must share the same copy of these data sets.

- **May be shared**

  There is no system requirement for these data sets to be either unique or shared (in a cloned IMS environment). There can be, however, other reasons (for instance, performance or data set management) why the user can choose to make them either unique or shared, or to provide clones (identical copies).

This is a very important part of the migration process, and it requires a good understanding of the uses of these data sets and the impact of the decision. This section discusses the characteristics and uses of the IMS and CQS data sets and some considerations for handling them in the target environment. Appendix B, "IMS System Data Sets" on page 215 contains a table of the data sets and some recommendations for their management.

### 5.2.1 IMS Data Sets

- **Must be unique**

  These are data sets that are updated by the IMS subsystems. Examples are the IMS logs, the RDS, and the message queue data sets.

- **Must be shared**

  Only the RECONs and the IMS database data sets fall into this category.

- **May be shared**

  Examples are ACBLIB, FORMAT, and program libraries.

## 5.2.2 CQS Data Sets

CQS data sets have the following requirements. They:

- **Must be unique**

  Each CQS must have a unique checkpoint data set for each shared queues structure pair. There can be up to two shared queues structure pairs, one pair for the full function message queues and one pair for the EMH queues.

- **Must be shared**

  The structure recovery data sets fall into this category.

- **May be shared**

  IMS.PROCLIB is an example.

## 5.2.3 Data Set Characteristics

Each data set has characteristics which should be considered in making these decisions.

***What does it contain?***

- Code
- Control blocks
- JCL
- Parameters
- IMS system data
- User data
- Other

Some data sets can contain more than one type of data. For example, IMS.PROCLIB often contains JCL and parameters. RESLIB contains both code and control blocks. The individual members might have different requirements for uniqueness or sharing.

***When is it used?:*** Data sets can be used in one or more of the following processes. The emphasis on the handling of data sets should be on those that are accessed by an active online system as opposed to those that are not, such as DBDLIB.

- Installation and maintenance

  These data sets are used for subsystem installation and maintenance, or as input to the IMS system definition process. (There is no system definition process for CQS.) Examples are the SMP/E, DLIB, and Target data sets.

  Keep in mind, when these libraries are shared, an implicit assumption has been made. That is, we have decided to implement a cloned environment, and in a cloned target environment, all subsystems are at the same version and maintenance level. This might not be practical when we consider how change is to be introduced to the Parallel Sysplex environment.

  When change is introduced, for example introducing a new maintenance level, a separate set of libraries which reflect the change are likely to be desirable to allow the introduction of the change to be accomplished in a round-robin fashion. Until the change has been introduced to all systems, two sets of these installation/maintenance libraries are a likely requirement.

The alternative, of course, is to bring all IMS systems down at the same time and restart them with the new set of libraries. This alternative is not recommended for two reasons. One, it incurs an unneeded outage of all systems to incorporate change, and, two, it exposes all of the restated IMS systems to unplanned outages introduced by the change.

- Customization

  These are data sets which are not accessed directly by an executing subsystem but are used to define or customize that subsystem. Examples are the DBD and PSB libraries, MFS referral libraries, various staging libraries, and intermediate data sets created during the IMS system definition process.

- Execution time

  These are accessed directly by an executing subsystem. Examples are the IMS and MVS logs, the ACB libraries, and the RECONs.

***What is the access intent at execution time?:*** For those that are accessed by an executing subsystem, what is the access intent of that subsystem?

- Read only

  An executing subsystem does not update these data sets.

- Update (read/write)

  An executing subsystem can read and write to these data sets.

- Write only

  An executing subsystem only writes to these data sets.

## 5.3 Executable Code

Executable code in a Parallel Sysplex IMS environment can be classified as IMS system code, CQS system code, exit code, or application code.

### 5.3.1 IMS System Code

The IMS system code is usually found in IMS.RESLIB. Many of the modules found in this library are common to all IMS subsystems. There are, however, some which are unique to each subsystem. The IMS system definition process identifies these by appending a suffix to the module name (such as, DFSVNUCx). Those that are common to all subsystems do not have the suffix and will be shared by all subsystems. The suffix itself is defined by the user in the IMSGEN macro:

```
IMSGEN ...,SUFFIX=x
```

This suffix, which is also used to distinguish control block modules generated by the system, would allow all subsystems to *share* the same RESLIB even though they are not identical clones. Whether or not to clone system definitions is an option an IMS installation has.

With IMS V6.1, IMS TM systems can be identical clones of one another and are thus able to share the same nucleus. Cloning assumes, however, that the installation wants to run all IMS subsystems at the same maintenance level and with the same system definition options. If this is not the case, then it will be necessary to define at least two RESLIBs.

The following scheme can be used to introduce maintenance to one system at a time. It uses two data sets for RESLIB. Data set IMS.RESLIB would be used by all systems where new maintenance is not being introduced. Data set IMS.MAINT.RESLIB would be used by those systems where new maintenance is being introduced. The procedures for the systems would include the following statement in the STEPLIB concatenation.

```
//STEPLIB DD DISP=SHR,DSN=IMS.&SYSM.RESLIB
```

For systems using the normal RESLIB, the procedure would be executed with a null value for SYSM. For systems using the new maintenance, the procedure would be executed with SYSM=MAINT.

Because there is relatively little access to RESLIB during online execution, it should not be necessary to clone it for performance reasons. That is, RESLIBs should either be unique or shared.

### 5.3.2 CQS System Code

Executable CQS code resides in RESLIB along with IMS executable code. There are no system definition requirements for generating executable CQS code. All IMS V6.1 systems are shared-queues-capable, which includes executable CQS code. The use of CQS with shared queues by an IMS system is an execution option on the IMS procedure (new keyword option SHAREDQ=xxx where xxx is the suffix for a new proclib member, DFSSQxxx). Therefore, all CQSs are able to share the same RESLIB. At a minimum, each CQS should share the same RESLIB with its associated IMS.

### 5.3.3 Exit Code

Exit routines, whether written by the user or provided by IMS, must be in libraries in the //STEPLIB concatenation for the address space that invoke them. This can be the control region, the DLISAS region, or even a dependent region (for example, the data capture exit). Exit routines can reside either in their own libraries or in the IMS RESLIB.

Some exit routines must be common to all IMSs (for example, database randomizers and compression routines) and should share a common library. Others can be tailored for a particular IMS and must be in a library which is unique to that IMS (for instance, found in the //STEPLIB for only that IMS).

CQS client and user exit routines are individually loaded and executed in the IMS control region. These CQS exit routines cam be shared among all IMSs in the Parallel Sysplex.

The following example shows an IMS with both a common and a unique exit library. Data set names containing the IMS subsystem name, PD11, are unique to that IMS subsystem.

```
//STEPLIB  DD  DSN=IMS.IMSP.PD11.EXIT,DISP=SHR
               DSN=IMS.IMSP.EXIT,DISP=SHR
               DSN=IMS.IMSP.PD11.RESLIB,DISP=SHR
```

See 16.5, “User Exits” on page 162 for a discussion of IMS exits.

### 5.3.4 Application Program Code

**TM Only**

This section applies only to IMS TM

Application program modules must be in a library in the dependent region //STEPLIB concatenation. Usually, these program modules are independent of the IMS system on which they execute and so can be in shared, cloned, or unique program libraries. One consideration, however, is the I/O that might be encountered loading these modules into the dependent region address space. If too many regions are reading the same library, the performance of this library might be degraded. The impact will depend, of course, on IMS′s use of preloaded modules or LLA/VLF.

## 5.4 Control Blocks

Control blocks define the IMS resources which are available to an IMS subsystem. They are generally created as a result of the IMS system definition process or one of the generation or service utilities (such as, ACBGEN or IMSDALOC). There are also control block libraries which are not accessed directly by the executing IMS subsystem but are used as intermediate or staging libraries for the creation of online control blocks (such as, DBDLIB, PSBLIB, and online change staging libraries).

### 5.4.1 CQS Access

CQS uses control blocks, but they are dynamically created when needed during execution. Therefore, this discussion of control block libraries does not apply to CQS.

### 5.4.2 Online Access

The Table 1 identifies the libraries (by DDNAME) which contain control blocks accessed directly by the online system.

*Table 1 (Page 1 of 2). IMS Online Region Library*

| TYPE REGION | DDNAME | CONTENTS | COMMENTS |
|---|---|---|---|
| CONTROL<br>DLISAS | STEPLIB | System defined resources (for example, databases, applications, transactions, network) | Created by system definition process (except MODBLKS); contained in suffixed modules in RESLIB, therefore can be identified for each IMS subsystem. |

*Table 1 (Page 2 of 2). IMS Online Region Library*

| TYPE REGION | DDNAME | CONTENTS | COMMENTS |
|---|---|---|---|
| CONTROL<br>DLISAS | MODBLKSA<br>MODBLKSB | Changes to system defined resources (except network) | Updated by online change utility (DFSUOCU0) from MODBLKS system definition staging library; contained in suffixed modules in MODBLKS library, therefore can be identified for each IMS subsystem. |
| CONTROL<br>DLISAS | STEPLIB<br>IMSDALIB | Dynamic allocation members. See the fix to APAR PQ12171 for a discussion of IMSDALIB. | Dynamic allocation utility. Some of these members must be the same for all IMS subsystems while others must be different. |
| CONTROL<br>DLISAS | ACBLIBA,<br>ACBLIBB | PSBs, DMBs | Online change utility (DFSUOCU0) from ACBLIB staging library |
| CONTROL | MATRIXA,<br>MATRIXB | Security tables | Online change utility (DFSUOCU0) from MATRIX staging library |
| CONTROL | FORMATA,<br>FORMATB | MFS formats | Online change utility (DFSUOCU0) from MFS FORMAT staging library |
| CONTROL | IMSTFMTA,<br>IMSTFMTB | MFS test formats | The MFS test format library is concatenated with FORMATA and FORMATB. When online change changes FORMAT libraries, the same test format library is used as the first library in the concatenation. |

## 5.5 Parameter Libraries

These libraries contain execution time parameters for IMS and CQS. Most of them are identified by a prefix (such as, DFSPB) and a suffix specified in the execution JCL (for example, // EXEC IMSA,RGSUF=AAA). IMS looks for members containing these parameters in the data set allocated by the //PROCLIB DD statement, usually IMS.PROCLIB. They are not PROCs, however, but PARMs and should be handled differently. For purposes of this document, we will refer to the library containing these parameters as IMS.PARMLIB and assume that it is concatenated with whatever else can be specified in the //PROCLIB DD statement.

## 5.6 Dynamic Allocation

Dynamic allocation members for an IMS subsystem reside in libraries which are part of the STEPLIB concatenation. Some of the data sets defined in these members will be shared. Others must be unique. Therefore, it will probably be necessary to have two sets of dynamic allocation libraries, one common library for shared data sets and one which is unique to each IMS for unique data sets. The STEPLIB would be similar to the following.

```
//STEPLIB  DD  DSN=IMS.IMSP.PD11.EXIT,DISP=SHR
               DSN=IMS.IMSP.EXIT,DISP=SHR
               DSN=IMS.IMSP.PD11.DYNALLOC,DISP=SHR
               DSN=IMS.IMSP.DYNALLOC,DISP=SHR
               DSN=IMS.RESLIB,DISP=SHR
```

In this example, the data set names with PD11 in them are used only by IMS subsystem PD11. The data set names without PD11 are used by all IMS subsystems.

If PD12 existed, its STEPLIB would be similar to the following.

```
//STEPLIB  DD  DSN=IMS.IMSP.PD12.EXIT,DISP=SHR
               DSN=IMS.IMSP.EXIT,DISP=SHR
               DSN=IMS.IMSP.PD12.DYNALLOC,DISP=SHR
               DSN=IMS.IMSP.DYNALLOC,DISP=SHR
               DSN=IMS.RESLIB,DISP=SHR
```

If multiple libraries are used, then be sure to update the DFSMDA utility JCL (//SYSLMOD DD) to put these members in the correct data set.

## 5.7 JCL Libraries

These are libraries which contain JCL for IMS execution or application execution. The JCL can be in the form of procedures (PROCs) or jobs. This section discusses only the libraries, not the actual members of the libraries. Refer to Chapter 17, “IMS and User JCL” on page 167 for that discussion.

### 5.7.1 Procedures

These libraries contain IMS- and CQS-related procedures (PROCs) which are either started tasks (for example, START IMSA) or executed procedures such as // EXEC DFSMPR. Each PROC must either be in SYS1.PROCLIB or be in a library which is concatenated to SYS1.PROCLIB (frequently IMS.PROCLIB). They are not executed from any library defined in either IMS or CQS subsystem execution JCL. IMS and CQS are, therefore, dependent on the operating system to execute the correct procedures.

some of these procedures must be unique to the system which uses them while others are common and can be shared. Because any IMS and its associated CQS can run on any MVS (with one SYS1.PROCLIB concatenation), these PROCs must be carefully named and stored so that there will be no problems when, for example, IMSA along with CQSA are started on MVSB rather than MVSA.

### 5.7.2 Jobs

These libraries contain JCL for jobs which are submitted through the MVS internal reader (INTRDR), usually by IMSRDR, as a result of the /START REGION command but also by DBRC as a result of the GENJCL command. In some cases, the JCL for the jobs are first stored in a job library and then submitted using the TSO SUBMIT command. IMSMSG and IMSWTnnn are examples of jobs which are usually found in these libraries, but the user can also include any job which is to be started using the /START REGION <member> command, the GENJCL.USER

MEMBER(name) command, or that is to be submitted using the TSO SUBMIT command.

Because the job being submitted is defined in the JCL of the procedure submitting the job, any library can be used as long as the submitting JCL specifies the correct library.

For example, the /START REGION <member> command executes the IMSRDR PROC, which in turn submits the job specified in the <member> parameter.

```
/START REGION MSGA01  <-------------- MTO COMMAND

S IMSRDR,MBR=MSGA01   <-------- Internally issued START IMSRDR

//IMSRDR   PROC MBR=MSGA01,...  <---- IMSRDR PROC from PROCLIB
//IEFPROC  EXEC PGM=IEBEDIT           with MBR overridden
//SYSUT1   DD DDNAME=IEFRDER
//SYSUT2   DD SYSOUT=(&CLASS,INTRDR),DCB=BLKSIZE=80
//IEFRDER  DD DISP=SHR,DSN=IMS.JOBS(&MBR)  <---  Contains MSGA01 job
```

There is a parameter in the IMS control region (PRDR) which allows the user to specify an alternative name for IMSRDR (such as, PD11RDR). Therefore, each IMS can have its own unique IMSRDR PROC specifying a unique IMS.JOBS library (for example, IMS.IMSP.PD11.JOBS) which contains jobs that are unique to that IMS.

```
/START REGION MSGP01

S PD11RDR,MBR=MSGP01

//PD11RDR  PROC MBR=MSGP01,...
//IEFPROC  EXEC PGM=IEBEDIT
//SYSUT1   DD DDNAME=IEFRDER
//SYSUT2   DD SYSOUT=(&CLASS,INTRDR),DCB=BLKSIZE=80
//IEFRDER  DD DISP=SHR,DSN=IMS.IMSP.PD11.JOBS(&MBR)
```

IMS.IMSP.PD11.JOBS would have to have a member named MSGP01, which could be tailored specifically to PD11.

```
//MSGP01  JOB   .....
//        EXEC  MPRP01,IMSID=PD11 <--- JOB tailored for PD11


//MPRP01  PROC  ....             <--- PROC common to all IMSs
```

In this way, a common IMSRDR PROC can be used to execute jobs from IMS.JOBS libraries that are unique and tailored to a specific IMS regardless of which MVS system it is running on.

With IMS V6 the /START REGION command has been enhanced and gives us the ability to share IMS.JOBS library members among different IMS subsystems. This eliminates the need for a unique IMS.JOBS library for each IMS in the Parallel Sysplex. This is made possible by the following:

- Include dependent region execution procedures in each control region's //PROCLIB DD concatenation (conceivably, this could be the IMS.JOBS library). The execution procedure found in the //PROCLIB DD concatenation is used when a /START REGION command is entered that includes one or both of the new keyword options with the entered command and are described in the bullets that follow. When one or both of these new keyword options is used, the IMSID= keyword option in the procedure is overridden with the IMSID of the IMS subsystem on which the /START REGION command was entered.

- Overlay the jobname of the IMS.PROCLIB procedure member with the jobname specified with the entry of the /START REGION command. For example, the entry of /START REGION MPRX JOBNAME MPR3 on IMSA results in the start of an MPR under IMSA whose jobname is MPR3.
- Allow a new keyword option of LOCAL to be entered with the /START REGION command. When LOCAL is specified, the symbolic IMSID parameter in the JCL for a dependent region is overridden with the IMSID of the system on which the command was entered. The LOCAL option is the default if the JOBNAME keyword is also specified.

The ability to override IMSID= and the jobname specified in the dependent region execution procedure allows a single procedure to be shared among multiple IMS systems.

When either of the new keywords is not used with the /START REGION command, the dependent region JCL to be executed is read from the data set pointed to by the //IEFRDER DD statement in the IMSRDR procedure. This allows a user to start dependent regions in the same manner as today which was discussed at the start of this section.

## 5.7.3 DBRC JCL Libraries

The DBRC GENJCL command uses two libraries.

### 5.7.3.1 Skeletal JCL

This is the source of skeletal JCL for the GENJCL command. It is defined in the DBRC PROC by the //JCLPDS DD statement.

```
//JCLPDS   DD  DSN=.....
```

The skeletal JCL for GENJCL.ARCHIVE includes a STEPLIB pointing to a RESLIB. However, since it is only used to find a library with the ARCHIVE program (DFSUARC0), it probably does not matter which RESLIB is identified. And, since the data sets have DSNs which include .%SSID. qualifiers, there should be no reason why it is necessary to have unique libraries.

### 5.7.3.2 GENJCL Output

This library contains the jobs which result from the GENJCL command. The library is defined in the DBRC PROC by the //JCLOUT DD statement. Since this is an output data set, each IMS subsystem would require its own data set.

```
//JCLOUT   DD  DSN=IMS.IMSP.PD11.JCLOUT,DISP=SHR
```

Alternatively, JCLOUT can specify the JES internal reader, in which case the JCL is submitted as soon as it is created, and there is no JCLOUT library.

```
//JCLOUT   DD  SYSOUT=(A,INTRDR)
```

## 5.8 Making the Decision

For some data sets, the user might choose whether to share them or to provide unique data sets for each IMS system. For others, there is no choice. For example, the RECONs must be shared, the LOGs must be unique, and the ACB libraries can be shared or unique.

The following identifies most of the IMS system data sets and groups them in the following categories.

- Must be unique
- Must be shared
- Probably shared
- Probably unique

Appendix B, “IMS System Data Sets” on page 215 summarizes this information in tables.

### 5.8.1 Data Sets That Must Be Unique

Usually, if a system data set is updated (written to) by IMS or CQS, each must have its own version of the data set. Two exceptions to this statement are the structure recovery data sets (SRDSs), which are shared among the CQS members of a shared queues group, and the RECON data sets, which are shared among all of the IMSs in a data-sharing group. The following list identifies those IMS or CQS system data sets which must be unique. Unless specifically identified as a CQS data set, the data set is assumed to be an IMS system data set.

- IMS Logs

  This includes OLDSs, RLDSs, SLDSs, and WADS.

- Message Queues *(IMS TM only)*

  This includes Queue Blocks, Short Message, and Long Message data sets. These data sets are only required if traditional queuing is used.

- CQS checkpoint data set(s). In a shared queues environment, each CQS requires a unique checkpoint data set for each shared queue structure pair.
- Restart Data Sets (IMSRDS)
- Modify Status (MODSTAT)
- IMS Monitor (IMSMON)
- External Trace (DFSTRA0n)
- DBRC JCLOUT
- MSDB *(IMS TM only)*

  This includes MSDB checkpoint, dump, and initialization data sets.

- SYSOUT *(IMS TM only)*

  These are IMS TM data sets defined in LINEGRP macros.

### 5.8.2 Data Sets That Must Be Shared

There is only one set of IMS system data sets that must be shared by all IMS subsystems in the data-sharing group.

- RECONs

  These data sets, which are accessed and updated only by DBRC, contain information about databases which **must be common** to all IMSs in the data-sharing group. Although each IMS has its own DBRC address space, each DBRC address space must use the same set of RECONs. Their integrity is ensured by DBRC.

With shared queues, the various CQS subsystems share

- Two pairs of structure recovery data sets (SRDSs). One pair is used to hold structure checkpoints (most recent and second most recent structure checkpoints) of a full-function message queue structure pair and the other the structure checkpoints of an EMHQ structure pair.
- MVS logger offload data sets for each shared queue structure pair. MVS logger offload data sets are associated with up to two user-defined MVS logstreams. One logstream is required to support a full-function message queue structure pair and the other to support an EMHQ structure pair.
- MVS logger staging data sets are optional and must be shared if used. The use of these data sets is recommended when the Coupling Facility where the logger structures reside becomes volatile.

### 5.8.3 Data Sets That Are Probably Shared

The following data sets can either be shared by all IMSs, cloned for each IMS, or each IMS can have its own unique version. They are, however, categorized by the probability of being shared.

- Probably shared at execution time

  The following data sets will probably be shared by cloned systems. They are accessed by executing IMS systems, but not modified.

  - ACBLIB, ACBLIBA, and ACBLIBB

    ACBLIB is the staging library. ACBLIBA and ACBLIBB are read by executing IMS systems. *It is required that DMBs for a shared database be the same.* For cloned systems, the PSBs used by each system should be the same. For these reasons, it is assumed that one instance of each of these data sets will be used by all clones.

  - FORMAT, FORMATA, FORMATB, IMSTFMTA, IMSTFMTB *(IMS TM only)*

    FORMAT is the staging library. FORMATA, FORMATB, IMSTFMTA, and IMSTFMTB are read by executing IMS TM systems.

  - RESLIB

    RESLIB contains most of the IMS executable code. Each IMS system has its own nucleus module which is identified by a unique suffix. Since most of the code is loaded at initialization time, RESLIB is not accessed often during normal executions. It is unlikely that unique copies would be required for performance reasons. If one RESLIB is shared, all IMS systems will be at the same maintenance level and have the same defined features.

If an installation wants to be able to run different IMS subsystems at different maintenance levels, they must have different RESLIBs. This is especially desirable when new maintenance levels are introduced. It might be advisable to implement the new level on one system instead of all systems simultaneously. For installations with continuous availability (24x7) requirements, this will allow them to introduce the new levels without bringing down all systems at the same time.

  - MODBLKS, MODBLKSA, and MODBLKSB

    These data sets contain control blocks and are used by online change. They are associated with a RESLIB. If there are multiple RESLIBs, there should be multiple MODBLKS, MODBLKSA, and MODBLKSB data sets.

  - MATRIX, MATRIXA, and MATRIXB

    These data sets contain security tables and are used by online change. They are associated with a RESLIB. If there are multiple RESLIBs, there should be multiple MATRIX, MATRIXA, and MATRIXB data sets.

- Probably shared, but not at execution time

  The following data sets will probably be shared by cloned systems. They are not accessed by executing IMS systems.

  - REFERAL *(IMS TM only)*

    This data set is used by the IMS TM MFS Language utility.

  - LGENIN and LGENOUT

    These data sets are used during the system definition process when the Large System Generation (LGEN) option is specified.

  - OBJDSET

    This data set contains assembler output created during system definition stage 2.

  - OPTIONS

    This data set contains the configuration dependent macros stored by system definition stage 2.

- Assumed to be shared

  The following data sets are assumed to be shared by all IMS systems at the *same release and maintenance level*. These data sets are used for installation, maintenance, system definition, and some utilities (such as, DBDGEN, PSBGEN, ACBGEN, and DFSMDA).

  If multiple maintenance levels are desired, it will be necessary to have multiples of these data sets.

  - SMP/E, DLIB, IVP, and some Target data sets

    This includes DLIBZONE, GLBLZONE, TRGTZONE, SMPMTS, MPTPTS, SMPSCDS, SMPSTS, GENLIB, GENLIBA, GENLIBB, LOAD, DBSOURCE, TMSOURCE, SVSOURCE, DFSCLSTA, DFSMLIBA, DFSPLIBA, DFSTLIBA, DFSEXECA, DFSISRCA, DFSRTRMA, SVOPTSRC, DBOPTSRC, TMOPTSRC, DFSCLST, DFSMLIB, DFSPLIB, DFSSLIB, DFSTLIB, DFSEXEC, DFSRTRM, DFSISRC, and MACLIB.

### 5.8.4 Data Sets That Are Probably Unique

If an installation wants to be able to run different IMS subsystems at different maintenance levels, they must have different RESLIBs and probably will want different MODBLKS, MODBLKSA, MODBLKSB, MATRIX, MATRIXA, and MATRIXB data sets. See the discussion above under ″Data Sets Which Are Probably Shared″ for these considerations.

**For IMS TM Only**

The library with DDNAME of DFSTCF contains Time-Controlled Operations (TCO) scripts. TCO is optional and can only be used with IMS TM. Since IMS initially loads the member with name DFSTCF, systems with different needs will require different libraries.

# Chapter 6. Applications and Databases

The target environment consists of multiple IMS subsystems running on multiple MVS systems, configured in a single data-sharing group within the Parallel Sysplex. The IMS configuration may consist of cloned, partitioned, or joined IMS subsystems, or may be a hybrid (combination).

**CLONED** Each IMS system is a ″clone″ of the other(s). This means that all applications and all databases are available to any IMS subsystem in the data-sharing group.

**PARTITIONED** Each IMS system runs its own subset of applications, but some or all of the databases can be shared. This term is most often used when an installation begins with one IMS system and splits it into multiples.

**JOINED** This is another term for PARTITIONED. This term is most often used when an installation begins with multiple IMS systems and adds sharing of their databases.

**HYBRID** This is a combination of the above, with some applications running on only one IMS subsystem (PARTITIONED or JOINED) and others running on multiple IMS subsystems (CLONED).

While the underlying assumption in this document is that the existing IMS is to be cloned, it is also understood that some installations might choose to partition applications or databases.

## 6.1 Applications

Most applications will be cloned. That is, a program which runs on IMSX now will run on both IMSA and IMSB in the target environment. There can be some applications, however, which will not be cloned. Instead, they will be partitioned and run only on IMSA or on IMSB. Some of the reasons why an application might run on only one IMS are:

- Performance would be degraded in a data-sharing mode, and the capacity and availability benefits of cloning are not required for the application. For example, the application might be a heavy updater of a small database and the buffer invalidation overhead would be undesirable.
- The application can access a non-sharable resource, such as an MSDB not converted to a DEDB using VSO.
- The application′s transactions must run serially (for example, FIFO). These will be defined in the IMS TM system definition as follows:

  ```
  TRANSACT ...,SERIAL=YES     (serializes transaction scheduling)
  ```

  See the discussion on handling serial transactions in the section 10.6, “Serial Transactions” on page 102.

These PSBs and transactions must be identified and provision made for routing them to the correct IMS system.

## 6.2 Databases

In general, database data sets do not need special consideration. By definition, they will be shared in a cloned IMS environment. However, by choice, some databases might not be shared. For example, an installation might not convert an MSDB, which is not sharable, to a DEDB using VSO, which is sharable. Each database which is not to be shared must be identified.

## 6.3 Partitioned Applications and Databases

Although the term ″affinities″ is most often used with CICS systems, there can be some affinities within an IMS environment. Some examples are:

- If a database is non-sharable, such as an MSDB, then users of that resource must execute on the one IMS system which has access.
- If a table is maintained in memory and accessed by application programs, all such programs must execute on the same system.

Those applications and databases which are not or cannot be cloned (for instance, they are partitioned) must be identified and provisions made to provide accessibility. For example, if the PERSONNEL application and its databases are not to be cloned, then users of the personnel application must have access to that IMS which processes the personnel transactions. This accessibility can be handled by shared queues, IMS MSC, CICS transaction routing, or other means. If a particular database is not shared, then all users (such as, IMS subsystems, transaction codes, and application programs) of that database must be identified for partitioning.

## 6.4 Cloned Applications and Data Sets

Applications can have non-database data sets or unshared databases. If these applications are cloned, provisions for handling these data sets and databases must be made. If they are used for input to applications, copies of the data sets and databases might be needed for each system. If they are created or updated by the applications, the data from multiple systems might need to be combined. The handling of these copies will typically be done by batch processes.

Non-database data sets and unshared databases must be identified. If copies of them are needed for input to the cloned systems, a process for creating the copies must be developed. If the clones produce different versions of these data sets, a plan for consolidating them must be developed.

## 6.5 Handling Databases That Are Not Shared

If some databases are not shared, the following techniques can be used.

### 6.5.1 Routing Transactions

Both IMS Transaction Manager and CICS have facilities for routing transactions between systems. IMS TM′s shared queues support, IMS TM′s Multiple Systems Coupling (MSC), CICS′s dynamic transaction routing, and Intersystem Communication (ISC) for both IMS TM and CICS allow a system to route transactions to other systems. Unshared databases can be assigned to one IMS database system, and all transactions requiring access to these databases can be routed to an IMS TM or CICS system with access to these databases.

See also the discussion in the section 9.3.2, “MSC Balancing” on page 82.

### 6.5.2 Copying Databases

Some databases are rarely updated. Copying these databases can be appropriate in a cloned environment. An example is an MSDB used primarily for query purposes. Providing copies of these databases to each of the sharing subsystems can be acceptable. If they are updated by a batch process, the update process will have to be modified so that the process is applied to each copy. If they are updated by an online transaction, the transaction will have to be invoked on each sharing subsystem.

# Part 3.  Planning Considerations for IMS TM

# Chapter 7. Introduction to IMS TM Considerations

**TM Only**

This chapter applies only to IMS TM

Although not an official IBM term, this publication uses the term *IMSplex* to represent a collection of IMS TM subsystems that are related, either by sharing common databases and/or message queues or by providing common services to the end-user community. In effect, the end-user should view the IMSplex as a single entity, a provider of IMS service. An IMSplex is generally defined as those IMSs within the same Parallel Sysplex environment which are related in some manner to provide IMS services to end users in session with one or more of these IMSs. The IMSs within the IMSplex are related by being members of the same data- sharing and/or shared queues groups or by sharing workload using MSC or ISC.

While an IMSplex should have the appearance of a single provider of service to the end-user community, the IMS/ESA Transaction Manager implementation of IMSplex using traditional queuing is really just a collection of independent IMS systems that happen to be sharing databases and/or are interconnected through MSC or ISC links. Each IMS system has its own set of master terminals, its own set of message queues, its own set of message regions, and so forth. In other words, each of the IMS TM systems is an entity unto itself. The only things that tie the IMS TM systems together are data sharing (they share a common set of databases) and/or MSC/ISC links (they can route work from one system to another under application, user exit, or system definition control).

## 7.1 Overview

IMS/ESA Version 6 Release 1 with shared queues alters the preceding description. With shared queues, there is no need for a unique set of message queues; the queues are shared among all of the IMSs in the shared queues group. The shared queues replace the need to use MSC or ISC for routing work among systems.

We now discuss the various factors that must be taken into account, along with suggested solutions, in order to implement an IMSplex that comes as close as possible to achieving the installation′s objectives.

In general, one could describe the objectives of an IMSplex as:

- Present a single image (provider of service) to the end-user community.
- Provide greater availability and/or capacity to the end-user community.
- Allow the IT organization to take advantage of new technology with improved price/performance characteristics.
- Allow existing applications to function without change.

## 7.2 IMS TM Configuration Considerations

There are many ways to configure an IMS TM IMSplex. One can replace an existing IMS TM system with multiple IMS TM systems, each capable of providing the same services and functions as the original IMS TM system. This is called **cloning**. When cloning is used, the IMSplex contains multiple *servers*, each able to provide similar functions to the end-user community.

One can combine multiple, somewhat independent, IMS TM systems into an IMSplex. The systems to be moved to the IMSplex are not clones of one another. This situation might arise when these multiple IMS systems need to share a subset of the enterprise′s databases. The combining of multiple IMS systems is called **joining**. When joining is used, the IMSplex is not viewed as a single provider of service to the end-user community. Rather, it continues to be viewed as multiple, independent providers of IMS service. Of course, one can clone the independent IMS TM systems that have been joined.

Another configuration option is to replace an existing IMS TM system with one or more *front-end* and *back-end* systems.

With *traditional queuing* the front-end IMS systems have the IMS network connected to them but do not have any dependent regions to process application programs. The back-end IMS systems have dependent regions to process application programs but do not have the IMS network connected. Transactions and messages flow between the front-end and back-end systems through MSC and/or ISC links.

Front-end and back-end configurations can also be implemented using *shared queues*. As previously stated, the front-end systems have the IMS network connected to them, but do not have any dependent regions to process application programs. Transactions received from the network by the front-end systems are passed to the back-end application processing systems through the shared queues. Similarly, messages generated by the back-end application programs are placed on the shared queues where they are retrieved by the front-end systems and delivered to the terminals in the network. The use of shared queues replaces the need for connecting the front-ends and back-end systems with MSC and/or ISC.

A more in-depth discussion of these various configuration options is provided below.

### 7.2.1 Cloning

As stated earlier, cloning provides multiple IMS servers (IMS TM systems), each of which is capable of providing equivalent service to the end-user community.

The advantages of cloning are:

- Increased availability

  If multiple servers exist, the loss of one server does not cause complete loss of IMS service. Other IMS TM servers are still available to process the workload. In addition, preventive or emergency maintenance, either hardware or software, can be introduced into the IMSplex a server at a time.

- Increased capacity

As the workload requirements of the enterprise grow, cloning allows additional IMS systems to be brought into the IMSplex, as required, in a nondisruptive manner.

- Ability to benefit from new hardware technology

  Distributing the workload across multiple processors in the IMSplex, while preserving the image of a single IMS system to the end-user community, allows the enterprise to gain the benefits (price/ performance) of new processor technology without having to be overly concerned about the power of a single processor.

The concept of cloning is very simple. The implementation may or may not be simple. Cloning an existing IMS TM system into multiple IMS TM systems results in multiple IMS TM systems. These multiple IMS TM systems could be very similar, but, perhaps, not identical. Factors that might force systems to differ depend on several conditions:

- Whether IMS/ESA V5 or V6 is used. IMS/ESA V6 provides many functions and capabilities that allow cloning to be achieved as opposed to the use of IMS/ESA V5. For example:
  - IMS/ESA V6 allows data sharing of DEDBs with SDEPs and DEDBs with the VSO option. These two forms of a DEDB cannot participate in data sharing when IMS/ESA V5 is used.
  - IMS/ESA V6 allows the Master and Secondary Master Terminal definitions to be overridden at execution time. Therefore, all IMS/ESA V6 systems can be genned with the same Master and Secondary Master Terminal definitions. If IMS/ESA V5 were used, unique system definitions are required for the master and secondary master terminals.
- Whether traditional queuing or shared queues is used with IMS/ESA V6, Communication among IMS systems which use traditional queuing (for example, to achieve workload balancing) can be achieved using MSC and/or ISC. These MSC and/or ISC definitions must be different and, therefore, cannot be cloned.

  Distributing workload among IMS systems can be automatically achieved with shared queues, thus eliminating the need for unique MSC and/or ISC definitions.

If shared queues are to be used, it is likely that a single system definition can be used for all IMSs within the Parallel Sysplex. This is true even if a front-end/back-end implementation is desired. With IMS/ESA V6 and shared queues, the question should be *Why do I not want to clone my IMS systems?* rather than *What prevents me from cloning the IMS systems?*

### 7.2.2 Joining

The concept of joining involves combining two or more independent IMS TM systems into an IMSplex. Typically, these independent IMS TM systems are not quite independent. Rather, they have a small subset of databases that are common, but the majority of databases do not need to be shared.

An example of joining could be a situation where there are three IMS TM systems on three different MVS systems, each one handling the IMS processing requirements for three different manufacturing plants. The vast majority of the databases in each system are independent (unique to the specific manufacturing facility). However, there are a few databases that represent corporate, rather than plant, data. Prior to the advent of the Parallel Sysplex and n-way data sharing, each of the IMS systems had its own copy of the corporate databases, and the synchronization and maintenance of these databases was done through application logic on all three systems. For example, when one system updated its copy of the corporate data, it would send transactions to the other systems to update the other systems′ copies.

With the advent of n-way sharing, all three systems can be joined together in an IMSplex and share a single copy of the corporate data, thus eliminating the requirement for application maintained data synchronization. Nevertheless, after joining the three systems in a data-sharing environment in the prior example, the result is three unique IMS TM systems viewed by the end-user community as three unique IMS TM systems.

With *traditional queuing* and after joining the three IMS TM systems into an IMSplex, there are still three unique IMS TM systems, viewed by the end-user community as three unique IMS TM systems.

With *shared queues* and after joining the three IMS TM systems into an IMSplex, a single system image can be achieved from the perspective of the end users. VTAM Generic Resources can be used, for example, to balance the network workload across the joined systems while shared queues enable the transaction workload to be routed to the correct system for processing.

The joining of IMS TM systems is fairly straight forward from an IMS TM perspective. Only IMS data sharing and shared queue factors need to be considered.

If systems are to be joined rather than cloned, one gives up several advantages of a Parallel Sysplex environment. For example:

- Application availability. When an application can only execute on a single system, the loss of that system stops all processing for that application until the failed system is restarted.
- Workload balancing. By restricting an application′s processing to a single IMS, one cannot take advantage of excess processing capacity that might be available in the other IMS systems in the sysplex.

### 7.2.3 Front-End and Back-End

The concept of front-end and back-end IMS systems, connected through MSC links, is a fairly old concept and has been implemented by many users to address capacity and/or availability concerns. As stated earlier, a front-end/back-end configuration consists of one or more **front-end** IMS systems that are connected to the network and are responsible for managing message traffic across the network and one or more **back-end** IMS systems that are responsible for application (transaction) processing.

Front-end and back-end systems would typically be used in a data-sharing environment to solve a capacity problem where the resources available to a single IMS are insufficient to handle the workload.

*Traditional Queuing:* The front-end and back-end systems are normally connected through MSC links, but ISC (LU 6.1) links could be used if required for Fast Path EMH (Expedited Message Handler) reasons. The MSC links that interconnect the IMS TM systems within the IMSplex are referred to as

*Intra-IMSplex* MSC links. The MSC links that connect IMS TM systems within the IMSplex to other IMS TM systems outside the IMSplex are referred to as *Inter-IMSplex* MSC links.

*Shared Queues:* With shared queues, the MSC/ISC links connecting front-ends and back-ends can be removed and replaced by the shared queue structures in the Coupling Facility. Shared queues is an improvement over the use of MSC/ISC links to implement a front-end/back-end configuration as well as an improvement in addressing capacity and/or availability concerns. Lastly, the use of shared queues does not prevent the use of inter-IMSplex MSC/ISC links.

# Chapter 8. IMS TM Network Considerations

**TM Only**

This chapter applies only to IMS TM

This chapter discusses IMS TM network considerations.

## 8.1 Overview

Before the use of Parallel Sysplex, each IMS terminal is connected to a single IMS. This is also true in a Parallel Sysplex, although a terminal can be capable of logging on to any IMS in the IMSPlex. Of course, it will be logged on to only one IMS at any time. For example, in a single system, non-Sysplex environment, NODEA and NODEB might both be logged on to IMSX. In the target environment, it might not matter to which IMS a terminal logs on. On the other hand, it might be desirable for NODEA to be logged on to IMSA and NODEB to be logged on to IMSB. Such considerations are especially likely in systems which are not cloned.

The decision about how to connect the existing IMS network to the target IMS systems depends on load balancing, performance, and availability.

If applications are partitioned or execute in joined systems, terminals requiring access to those applications can be connected to the corresponding IMS for best performance whether traditional queuing or shared queues is used. For example, if the PERSONNEL application runs only on IMSB, it makes sense to connect the terminals in the PERSONNEL department to IMSB to achieve the best performance of the application.

Conversely, if applications and systems are cloned:

- Network workload balancing can be achieved through the use of VTAM generic resources.
- Application workload balancing is achieved through the use of shared queues.
- Improved availability is a result because the failure of one system does not eliminate all processing by an application.

A terminal network is defined to IMS either statically or dynamically. Dynamic definitions are created by the use of ETO. Both static and dynamic definitions can be used with an IMSplex.

## 8.2 Special Network Considerations

We now discuss a number of network connectivity considerations when an existing IMS system is cloned into two or more IMS systems within an IMSplex.

### 8.2.1 SLUTYPEP

SLUTYPEP is a program to IMS protocol defined by IMS (SNA LU 0). In addition to defined protocols for exchanging messages (entering transactions and receiving replies), the SLUTYPEP protocol also allows for resynchronization between the SLUTYPEP program and IMS following session and/or system failures. This resynchronization capability can be used to ensure that messages are not lost or duplicated upon a restart and is implemented using STSN logic. STSN, an SNA command, is the mnemonic for Set and Test Sequence Numbers.

When a SLUTYPEP terminal logs on to an IMS system, IMS sends an STSN command to the terminal informing the terminal as to what IMS remembered the last input message and last output message sequence numbers to be (the sequence numbers are maintained within both IMS and the SLUTYPEP program). The SLUTYPEP terminal uses this information to determine if it has to resend the last input and to inform IMS whether IMS should resend the last recoverable output message. This STSN flow occurs whenever a SLUTYPEP session is established with IMS, even if the session had been normally terminated the last time the terminal was connected to IMS.

There are two conditions where resynchronization cannot take place: When an IMS system is cold started (including a COLDCOMM restart), sequence numbers are forgotten by IMS and when using VTAM generic resource support with IMS and VTAM is requested to manage affinities (GRAFFIN=VTAM specified on the IMS procedure), sequence numbers are not remembered by IMS whenever a session is reestablished.

Thus, once a particular SLUTYPEP terminal has logged on to an IMS system within the IMSplex, it must continue to use that same system for all subsequent logon requests if the SLUTYPEP terminal is sensitive to the sequence numbers contained in the STSN command. However, if the SLUTYPEP terminal program is not sensitive to the sequence numbers contained in the STSN command (meaning it allows a session to be established whether sequence numbers are available or not), then the SLUTYPEP terminal would not have an affinity to the system with which it was last in session.

As discussed in 9.2.5, “Use of VTAM Generic Resources” on page 69, the choice of the type of VTAM generic resource affinity management for use with an IMS system is dependent upon whether the programs in the SLUTYPEP terminals depend upon synchronization of messages with IMS. If they are dependent upon message integrity, one must choose the execution option to allow IMS to manage affinities when VTAM generic resources are used. If the terminals are not dependent upon message synchronization, either IMS or VTAM can be selected to manage affinities.

## 8.2.2 ISC

ISC, or LU 6.1, is an SNA program-to-program protocol that allows IMS to communicate with “other” subsystems. ISC is normally used to send transactions and replies back and forth between IMS and CICS systems, but could also be used, in lieu of MSC, for IMS to IMS communications. Since LU 6.1 is an externally defined SNA protocol, IMS really does not know what type (CICS, IMS, other) of subsystem with which it is in session. As far as IMS is concerned, the other subsystem is a terminal.

There are a number of concerns that need to be addressed with ISC in an IMSplex environment.

### 8.2.2.1 Parallel Sessions

ISC supports VTAM parallel sessions. When parallel session support is utilized, all of the parallel sessions from another subsystem must be connected to the same IMS system. This might have an impact on USERVAR processing if one were to utilize that network balancing approach to IMSplex load balancing.

If VTAM generic resource support is used (log on to a generic resource name for IMS from a remote system), all parallel sessions are automatically routed to the same IMS system within the generic resource group by VTAM once the initial parallel session is established with a particular IMS in the group.

With VTAM generic resource support active, parallel sessions from remote systems can be established with a specific IMS in the group. If the first parallel session request is for a specific IMS VTAM APPLID, all subsequent requests for additional parallel sessions must use the same IMS APPLID as the first request. Otherwise, subsequent session requests will fail.

### 8.2.2.2 ISC Partner Definitions

The use of *traditional queuing* or *shared queues* results in different considerations and implications. The following discussion is split into two parts: ISC partner definitions when using traditional queuing and when using shared queues.

***ISC Definitions with Traditional Queuing:*** The following discussion refers to the use of static definitions within IMS to enable the establishment of ISC parallel sessions with the CHICAGO CICS system. ETO can be used to to dynamically create the control blocks required to establish and support these parallel sessions as well.

The cloning of an IMS system may or may not have an impact on ISC partners. Let us assume that an existing IMS system in Dallas (VTAM APPLID of DALLAS) has an ISC partner in Chicago (CICS system with a VTAM APPLID of CHICAGO). The Dallas IMS system is to be cloned into two IMS systems (VTAM APPLIDs of DALLAS and DALLAS2).

If a transaction executing on the second Dallas IMS system (DALLAS2) is to send a transaction to the CICS system in Chicago, one of two approaches can be used.

The definitions for the Chicago LTERMs could exist in both IMS systems as local LTERMs. In fact, all of the ISC-related definitions in both IMS systems can be the same. This alternative requires that the ISC resource definitions be changed in Chicago since the Chicago system needs to be connected to both IMS systems

(DALLAS and DALLAS2). This may or may not be an option. For example, if the Chicago CICS system belongs to a different enterprise, it might be difficult, if not impossible, to get Chicago to change their CICS resource definitions every time another IMS system in Dallas is added to the IMSplex.

An alternative is to leave the ISC-related definitions "as is" in the DALLAS IMS system and to define the ISC-related LTERMs as MSC remote LTERMs within the DALLAS2 IMS system. This results in all ISC-related message traffic to be routed through the DALLAS IMS system, and the cloning or splitting of the IMS systems is transparent to the Chicago system. This approach has two disadvantages:

- More work is performed within the Dallas IMSplex since messages destined for Chicago generated by transactions executing on DALLAS2 IMS system must be routed, through MSC, through the DALLAS IMS system. This MSC traffic may or may not be a problem (see 9.3.2, "MSC Balancing" on page 82).
- The DALLAS IMS system represents a single point of failure as far as connectivity between the Dallas IMSplex and Chicago is concerned.

It should be noted that the MSC routing exits, DFSCMTR0 or DFSNPRT0, are not invoked for transactions received from an ISC partner if the transaction code is stored in the ISC FMH.

***ISC Definitions with Shared Queues:*** The following discussion assumes the use of shared queues and uses the same example as before; that is, the DALLAS system is split into two IMS systems, DALLAS and DALLAS2.

If the original DALLAS ISC definitions to the CHICAGO CICS system were cloned, only the DALLAS IMS subsystem can establish a connection to the CHICAGO CICS system. This is true because the CHICAGO CICS system has no definitions that point to DALLAS2.

One alternative is to make no changes to the CHICAGO CICS system. That is, the CHICAGO and DALLAS systems have the active set of parallel ISC sessions. Message traffic from DALLAS2 to CHICAGO can be achieved through the shared queues and delivered to CHICAGO by DALLAS. Message traffic from CHICAGO to DALLAS2 also flows through the shared queues through the CHICAGO to DALLAS ISC connection.

The advantages of maintaining the ISC parallel sessions between CHICAGO and DALLAS are:

- The IMS systems in DALLAS can be clones of one another. This simplifies the SYSGEN process for DALLAS and DALLAS2. In addition, operational control of the connection between the Dallas IMSplex and the Chicago CICS system is unchanged.
- The routing of messages between DALLAS2 and CHICAGO through the shared queues is transparent to the application programs in all three systems.
- The CHICAGO CICS system definition does not have to change. As long as the DALLAS IMS system can handle the communication traffic, there is no need for considering implementing another connection from CHICAGO to DALLAS2.

One can consider defining and implementing another set of parallel ISC sessions between CHICAGO and DALLAS2 to remove the DALLAS to CHICAGO connection as a single point of failure. The considerations for doing this follow:

- The implementation of a connection between CHICAGO and DALLAS2 requires definition changes in both the CHICAGO CICS system (must add definitions to connect to DALLAS2) and DALLAS2 system (must define a new set of ISC definitions which incorporate LTERM names distinct from those in the DALLAS ISC definitions). The cloned ISC definitions from the original DALLAS ISC definitions must be discarded (cannot have two TERMINAL macros defined with the same partner nodename).
- The addition of another set of parallel sessions between CHICAGO and DALLAS2 requires that the application programs in the CHICAGO system be changed to split their originating message traffic between DALLAS and DALLAS2.

  The application programs within DALLAS2 might have to be changed. If DALLAS2 is acting as a pure back-end system (does not originate transactions to be sent to CHICAGO), the applications only require change if they are sensitive to the input IOPCB LTERM name. If the application programs in DALLAS2 originate transactions to be sent to CHICAGO, such as, insert transactions through an ALTPCB to be sent to CHICAGO, these must use the correct ISC LTERMname(s) to cause the transaction to flow across a DALLAS2 to CHICAGO parallel session.

### 8.2.2.3 Recovery

Like SLUTYPEP, the ISC protocol provides for session resynchronization between IMS and its partner subsystem following session and/or system failures. The VTAM STSN command and protocols are used for this purpose. Unlike SLUTYPEP, ISC session resynchronization is not required every time a session is established between IMS and its ISC partner.

An ISC session can be normally terminated through an IMS command such that session resynchronization is not required when restarted. When sessions are terminated using the /QUIESCE command or when the QUIESCE option is used with the shut down checkpoint command (/CHE FREEZE QUIESCE), a session, or sessions, are terminated without any sequence numbers in doubt, and resynchronization is not required upon session restart.

Another difference between SLUTYPEP and ISC is that ISC parallel sessions can be cold started without a cold start of IMS. The cold start of an ISC session simply means that sequence numbers required for session restart have been discarded. Keep in mind that the cold start of an ISC session might mean that messages between systems can be duplicated; no messages will be lost. The messages queued to an ISC session that is cold started are not discarded.

If ISC resynchronization is important, one should ensure that ISC partners always log back on to the same IMS system within the IMSplex following a session or IMS failure whether shared queues are used or not. This can be guaranteed by using specific VTAM APPLIDS in all system definitions between ISC partners or whenever a session request is made in an ETO environment. With two IMS systems in the IMSplex in an XRF configuration (one is the active and the other is the alternate), any system remote to the XRF configuration defines the USERVAR value in their definition.

## 8.3 APPC (LU 6.2)

This section discusses connectivity considerations for APPC (Advanced Program to Program Communication) partners using the APPC/IMS support introduced in IMS/ESA Version 4. Users of the LU 6.1 Adapter for LU 6.2 applications should reference the preceding ISC section since the adapter is defined as parallel ISC sessions to IMS.

The APPC/IMS support within IMS/ESA is built upon the APPC/MVS support provided by MVS/ESA. APPC/MVS provides the SNA session management support for APPC partners. Thus, an APPC partner logs on to a VTAM LU managed by APPC/MVS, not to the IMS APPLID. Conversations between IMS and an APPC partner are established with IMS upon request through the services provided by APPC/MVS.

APPC/MVS does support the use of VTAM generic resources to establish sessions between LU 6.2 programs remote to the IMS systems in an IMSplex that are members of the same generic resource group. An affinity between a remote LU 6.2 program and a specific IMS APPC LU exists whenever there is at least one protected conversation (SYNC_LEVEL= SYNCPT) active at the time of session termination. This affinity persists until the protected conversation is committed or backed out. See 9.2.6, “VTAM Generic Resources for APPC/IMS” on page 72 for information on specifying generic resources for APPC/IMS.

APPC/IMS provides two different levels of support for application programs. These are *implicit* and *explicit*.

The implicit support masks the APPC verbs and operations from the application program and preserves the “queued” system model for IMS. In effect, IMS places the transaction on the message queues, and the application program retrieves the message through a GU to the IO-PCB. Since the transactions and their responses flow through the IMS message queues, MSC facilities can be used to balance the APPC originated, implicit mode transactions among the various IMS systems within the IMSplex when traditional full-function queuing is utilized.

With shared queues, all transactions entered from an LU 6.2 application program must be processed in the front-end IMS if they are to be processed locally within the shared queues group. MSC facilities can be used to route LU 6.2 messages to IMS systems remote to the shared queues group.

The explicit support allows an IMS application program to be in direct APPC conversation with the APPC partner, thus giving the appearance of a “direct control” system model for the IMS application. An IMS application program using explicit support is responsible for issuing the APPC verbs and, as stated earlier, has a direct connection to the APPC partner. As such, the IMS message queues are bypassed, and MSC facilities cannot be used to balance the APPC-originated, explicit mode transactions among the various IMS systems within the IMSplex. Explicit mode transactions from a given APPC partner can only be processed by an application program running under the IMS system with which the conversation was started.

## 8.4 ETO Considerations with Shared Queues

There are many Extended Terminal Option (ETO) users. With ETO, of course, logical terminals ca be dynamically created. When ETO is used with a single IMS system, IMS ensures that duplicate logical terminal names are not created and, therefore, cannot be active at the same time. When ETO is used among the IMS members of a shared queues group, there are no system controls to ensure that the same logical terminal name is not active on two or more members of the same shared queues group. If this should occur, the results are likely to be interesting at best and unacceptable at the least. That is, responses to transaction input may be delivered to the wrong terminal (might be interesting, but definitely unacceptable).

Not every user of ETO is in danger of having the same logical terminal active on two or more members of the same shared queues group. Whether it can happen is dependent upon how the logical terminals are named. Listed below are ″ETO environments″ that cannot experience the problem:

- Implement a single front-end, IMS: With a single front-end, IMS system all active LTERMs are known within that front-end. Therefore, the front-end IMS ensures that all active LTERM names are unique.
- Implement a single front-end IMS with XRF: This configuration, is meant to address the shortcomings of a single front-end IMS. A single front-end IMS is a single point of failure. When XRF is added to the configuration, this single point of failure is removed.
- Each IMS uses a unique set of LTERM names associated with each user: This can be accomplished with (a) a unique set of nodename user descriptors for each IMS where the LTERM names specified on the descriptors are unique throughout the shared queues group, or (b) a Signon Exit in each IMS accesses a table of unique LTERM names where each user ID is associated with a unique set of LTERM names through the use of tables or through an algorithm (for example, unique suffix ranges are assigned to each IMS system).
- When a logical terminal name is always set equal to nodename or LUname, all logical terminals in session with the members of the same shared queues group have unique names because, by definition, all nodenames (LUnames) are unique.

### 8.4.1 Duplicate Terminal Names

Duplicate, active logical terminal names can occur, when ETO is used. The conditions under which it can occur are as follows:

- Multiple signons with the same user ID are allowed and
- Logical terminal names are set to the user′s user ID or some variation of the user ID.

For example, assume user IDS are seven characters or less in length. Unique logical suffix to the user ID. This works fine with a single IMS image because the Sign On Exit can ensure that the suffixes added to a given user ID are unique.

With shared queues, a given Sign On Exit can only check for suffixes that have been appended to a user ID in the IMS system under which it executes. Thus, it has no way of knowing what suffixes might have been ′used′ on other IMS systems that comprise the shared queues group.

### 8.4.2 Solutions

The following section discusses solutions to the user ID+suffix problem with multiple signons using the same user ID.

Assign suffixes to user IDs that ensures they are unique to a given IMS member of the shared queues group. For example, if a one character suffix is to be used and there are three IMS systems in the group, set up the Sign On Exit in each of the systems with a unique range of suffixes that can be added (IMSA can assign 1 through 3, IMSB 4 through 6, and IMSC 7 through 9). This solution ensures that a given user ID is always unique throughout the shared queues group.

The problem with this solution is the user can establish sessions with different IMSs as logoffs and logons occur. Establishing a session with a different IMS after a previous logoff is most likely if VTAM generic resource support is used to establish sessions. A solution to this concern is to use a VTAM Generic Resource Resolution Exit to always route a session request for a given nodename to the same IMS. This can be done by using a simple hashing algorithm (a simple division/remainder technique will work) on the nodename.

The use of a consistent hashing algorithm works well as long as all of the IMSs in the shared queues group are always available. When an IMS is added to or removed from the group, logical terminal output messages might become orphaned (never delivered) or might be delivered in an untimely fashion (long after they were queued).

Orphaned messages can be anticipated (they might occur whenever the number of IMSs in the shared queues group changes) such that corrective action can be taken (arbitrarily delete logical terminal messages from the shared queues after they have been queued for a period of time greater than some number of days). Or, orphaned messages can be avoided if the Generic Resource Resolution Exit rejects logon requests that would have been passed to an IMS system that is no longer active. This ′reject logons′ alternative is equivalent to not being able to log on to a single system that is not available for whatever reason.

The final alternative available is to change one′s logical terminal names from user ID+suffix to, for example, nodename. This last alternative ensures uniqueness of logical terminal names but does give up output message security.

The preceding discussion shows that there is not necessarily a good solution to the problem, but acceptance of the proposed solution does require an understanding of the potential severity of the problem if implemented. One must consider the answers to the following questions:

- With today′s IMS systems achieving 99+ percent availability, how often will the potential problem occur? Answer: Not very often.
- When the problem occurs, how many logical terminal queues will have output messages queued? Answer: Not very many.
- How often are we going to add or remove IMS subsystems to/from the shared queues group? Answer: Not very often.

Individual installations will make their implementation decisions based on the preceding discussion in light of their system needs and requirements. Keep in mind, when changing logical terminal names, application programs sensitive to the logical terminal name in the IOPCB can be affected.

## 8.4.3 Limiting the Number of End-User Signons

Some IMS users want to limit the number of signons to a single signon per user ID or to a specific number of signons with the same user ID. Once again, these limits are not difficult to enforce in a single system environment. Within a Parallel Sysplex environment, the number of times that a given user ID has signed on to the IMS members within the IMSplex is not known.

We are not aware of a solution for this problem.

## 8.4.4 Dead Letter Queue Considerations

*Dead Letter Queue (DEADQ)* is a status or attribute that is associated with an ETO user structure that has messages queued for delivery, but the user has not signed on for a specified number of days (DLQT= number of days as specified on the IMS execution procedure). Users with the DEADQ status are identified with the output from the `/DIS USER DEADQ` command. The DEADQ status only applies to ETO user structures. The attribute is not supported in a shared queues environment.

A new form of the display command (`/DIS QCNT qtype MSGAGE n`) has been implemented for use when using shared queues to identify by queue type (the queue types are TRAN, LTERM, BALGRP, APPC, OTMA, and REMOTE) those queue destination names that have messages queued that are older than the number of days specified (`MSGAGE n`, where `n` is the number of days) in the command. With shared queues, this new form of the display command applies to all destination names whether statically defined or dynamic (ETO).

In addition, the dequeue (`/DEQ`) command has been enhanced, when shared queues are used, to allow transactions to be dequeued.

The `/DIS QCNT qtype MSGAGE n` and `/DEQ` commands are meant to be the tools to assist in managing the accumulation of old messages on the shared queue structures. ′Old messages′ are not likely to be a problem for Fast Path EMH-driven input and output but are likely to require attention for traditional full-function shared queue messages. One thing to keep in mind, with shared queues is that an IMS cold start does not delete messages from the shared queues. An IMS cold start might very well have been used prior to the use of shared queues to ′manage′ message queue buildup. There are several alternatives that can be used to *manage* queue buildup when using shared queues:

- The messages on the shared queue structures can be deleted by deleting the structures on the coupling facility. This does require that all CQS address spaces be terminated before the structures can be deleted. IBM′s IMS Message Requeuer V3 (product number 5655-136) can be considered for use to restore selected queues after deleting the shared queues structures. Structure deletion, of course, requires an outage of all of the IMSs who are members of the same shared queues group.

  This alternative is viable when true continuous availability is not required and a periodic planned outage can be scheduled on a regular basis.

- Actively monitor and delete old messages from the structures using the commands described above to prevent the shared queues structures from becoming full. This monitoring and deletion of messages can be implemented using an automated operator program. Other automated operation tools can also be useful.

This second alternative is for those IMS users who want to achieve true continuous availability.

## 8.5 Conversational Transaction Processing

The SPA pool (pool size limit was specified with the SPAP= keyword on the IMS execution procedure in previous releases) has been eliminated with IMS/ESA V6. SPAs are now stored with the input and output messages associated with a conversation. The implications of this change are that storage is required elsewhere to store an SPA.

For an active conversation, the last output message and the SPA are stored in the shared queue structures as well as in the queue buffers of the IMS system that owns the conversation. When sizing the shared queue structures, allowance for this structure storage must be made. Allowance for SPA storage in the queue buffers is not necessarily a concern since the number of queue buffers can be expanded dynamically when shared queues are used.

Without shared queues, the SPA is also stored with the input and output messages associated with the conversation where the queue data sets are available to relieve any pressure that might occur because of the increased demand for queue buffers because they are now used to store the SPA.

For conversational users, a practical estimate of the shared queue structure storage required for SPAs can be made by including the size of pre-V6 SPA pool in the storage estimate for the shared queues. If traditional queuing is used, one can consider adding more queue buffers equivalent to the pre-V6 SPA pool size.

## 8.6 Application Program Considerations

Application programs do not have to be changed when implementing shared queues. The only difference that should be experienced is in the return from INQY ENVIRON calls. When shared queues are used, the characters ″SHRQ″ are placed at an offset of 84 bytes in the I/O area. Without shared queues, these four bytes are blanks. This field was introduced in IMS/ESA V6.1.

## 8.7 Application Affinities

IMS/ESA Version 6 eliminates most application affinities. All IMS databases, except MSDBs, can be shared. MSDBs can be handled by converting them to DEDBs using VSO. These can be shared. So, databases should not cause affinities. Nevertheless, some installations might have applications which either cannot be cloned or which they choose not to clone.

A BMP which serves as an interface between IMS and another technology may create an application affinity. An example is the *listener* BMP used for the IMS TCP/IP sockets support provided by MVS TCP/IP. If an installation wants only one TCP/IP sockets interface, only one copy of the listener BMP would be used.

## 8.8 IMS APPLIDs

When an existing IMS system is cloned into multiple IMS systems, the resulting IMS APPLIDs need to be unique (VTAM requirement). For example, if an existing IMS system with an APPLID of DALLAS were to be cloned into two IMS systems, only one (or none) of the resulting IMS systems could have an APPLID of DALLAS. The resulting IMS APPLIDs would either be DALLAS and DALLAS2, or DALLAS1 and DALLAS2. We would recommend that the old APPLID not be reused.

If one has a requirement that DALLAS still be known to VTAM, for example, changes external to IMS are difficult to make (that is, SLUTYPEP terminals issue VTAM INITSELF commands for DALLAS, remote ISC or MSC definitions specify an APPLID of DALLAS, workstation 3270 emulators have LOGON DALLAS hard coded, and so forth), we suggest that a VTAM USERVAR be defined as DALLAS and let VTAM transform the logon request to either DALLAS1 or DALLAS2. This will allow a much easier movement of workload when failure situations occur. Note, VTAM USERVAR usage is mutually exclusive with implementing IMS′s VTAM generic resource capability.

It can be common for an installation to have some terminals that must be "homed" to a particular IMS system and others that might be distributed among the various IMS systems within the IMSplex. A common case might be for SLUTYPEP terminals (ATMs) to be homed to a particular system while the rest of the network (3270s) can be distributed among all of the systems within the IMSplex. The SLUTYPEP terminals can use INITSELF to log on to a specific system, and the 3270 terminals use VTAM USSTAB processing to log on (that is, they would enter DALLAS on a cleared screen, rather than LOGON DALLAS). Let us assume that the desired system for the SLUTYPEP terminals is DALLAS2 (the APPLIDs of the two cloned systems are DALLAS1 and DALLAS2).

In this situation, one defines a VTAM USERVAR of DALLAS whose contents are set to DALLAS2. This setup satisfies the SLUTYPEP requirement.

The 3270 logon requirement is satisfied by performing two actions:

1. Define a USERVAR of DALLASP (for Dallas IMSplex). USERVAR DALLASP is to be used to distribute the logon requests between DALLAS1 and DALLAS2 (see 9.2, "Network Workload Balancing" on page 66).
2. Change the USSTAB entry for DALLAS from

```
    DALLAS  USSCMD  CMD=LOGON
            USSPARM PARM=APPLID,DEFAULT=DALLAS
              .
              .
              .
```

   to

```
    DALLAS  USSCMD  CMD=LOGON
            USSPARM PARM=APPLID,DEFAULT=DALLASP
              .
              .
              .
```

In this manner, the logon requirements are met without requiring any changes to the SLUTYPEP programs or the end user (3270) logon process.

## 8.9 VTAM Model Application Program Definitions

IMS TM users will find it advantageous to use VTAM model application program definitions in a Parallel Sysplex. These definitions provide two capabilities. First, they allow one VTAM definition to be used for multiple IMS systems. Second, they make it easy to open a VTAM ACB on any VTAM in the sysplex.

Model application program definitions are created by specifying one or more wildcard characters in the name field of the APPL definition statement in VTAMLST. An asterisk (*) is used to represent zero or more unspecified characters. A question mark (?) is used to represent a single unspecified character. Any ACB whose names can be built from these specifications can be opened when the model application program is active. The same model application program can be active on multiple VTAMs simultaneously.

Without the use of model application program definitions, moving an IMS system between systems requires operator commands. Commands must be issued to deactivate the program major node on one VTAM and activate it on another. Without model application program definitions, a program major node can be active on only one VTAM at a time. This restriction does not apply to model definition statements. They can be active on multiple VTAMs simultaneously.

## 8.10 A Model Application Program Definition Example

The following example illustrates a possible use of a model application program definition. The definition in VTAMLST is:

```
IMSPACB*  APPL AUTH=(PASS,ACQ,SPO),PARSESS=YES
```

The ACBs used by the various IMSs are:

```
IMSPACB1, IMSPACB2, and IMSPACB3
```

The major node in which the model application program is defined is in the configuration list that each VTAM uses when it is initialized.

One APPL statement is used for all of the IMS systems. This has two advantages. First, this definition allows any of the IMS systems to open its VTAM ACB on any VTAM. Second, moving an IMS system from one MVS to another does not require a VTAM VARY NET,INACT command to inactivate a major node on one VTAM and a VARY NET,ACT command to activate this major node on another VTAM.

# Chapter 9. IMS TM Workload Balancing Considerations

**TM Only**

This chapter applies only to IMS TM.

This chapter discusses a number of alternatives that might be used to balance the workload among the various IMS/ESA TM systems within a cloned IMSplex.

## 9.1 Overview.

There are two type of queues in IMS: traditional queues and shared queues.

*Traditional Queuing:* With traditional queuing, the IMS TM systems within the IMSplex are, for the most part, individual, independent IMS TM systems. When work enters one of these IMS systems from an end-user connected (logged on) to the IMS system, that work is processed within that IMS system unless the installation does something that would cause it to be routed (for example, through intra-IMSplex MSC links) to another IMS system within the IMSplex.

*Shared Queues:* With shared queues, the IMS TM systems are not independent with regard to the message queues. A transaction entered on one system can be processed by any IMS within the shared queues group that has interest in the transaction. All that is required is to clone transaction definitions among the IMSs within a shared queues group.

There are two types of workloads which require management among the IMS systems within a Parallel Sysplex environment. These are the network (terminal handling) workload and application (transaction processing) workload. These workloads must be distributed across the IMSs to achieve one′s objectives.

The distribution of these workloads across the IMS subsystems in the Parallel Sysplex environment is greatly dependent upon the hardware available and the selected configuration of the IMS subsystems that are to execute using this hardware. The possibilities include the following:

- The IMS systems can be clones of one another, and all IMSs execute using equivalent hardware resources. In this case, one can choose to distribute the network and application workloads evenly across all of the cloned subsystems.
- The IMS systems can be partitioned by application. In this case, distribution of the application workload is predetermined. All of the transactions for a given application must be routed to those IMS subsystems where the application resides. Two decisions that must be made with a partitioned configuration are to decide (a) what applications are going to execute where and (b) how is the network workload to be divided among the partitioned systems.
- The IMS systems can be divided into DC front-ends and DB back-ends. This decision just divides the network workload and application workload among two groups of IMS subsystems; the front-ends will handle the network workload and the back-ends the application workload.

The distribution of the network and application workloads need not be a complex exercise, but the actual distribution of work can be dependent upon a whole host of factors and considerations as the preceding, albeit incomplete, list of considerations was meant to show. We recommend that workload distribution be achieved in as simple a fashion as possible.

The concept of network workload balancing is to distribute end-user logon requests among the various IMS systems in the IMSplex such that the desired network workload balance is achieved.

The concept of application workload balancing is to route transactions among the various IMS systems in the IMSplex such that the desired application workload balance is achieved.

With **traditional queuing**, MSC can be used to route the transaction workload among the various IMSs in the IMSplex such that the desired application workload balance is achieved.

The use of **shared queues** introduces an entirely new set of application workload balancing considerations. With shared queues, the processing of an input transaction is, or can be, biased in favor of the IMS system where the input transaction was entered.

## 9.2 Network Workload Balancing

As stated earlier, network workload balancing attempts to balance the network workload among the IMS systems in the IMSplex by balancing the logon requests among the IMS systems. Depending upon how network workload balancing is achieved, your IMSplex configuration (cloned, partitioned, and so forth), and your workload balancing objectives, application workload balancing may or may not be an automatic result. If application workload balancing objectives are not achieved, additional implementation choices must be considered and selected to ensure the resulting application workload distribution satisfies your objectives.

There are several implementation choices to achieve network workload balancing, including:

- Instructions to the end (terminal operators) users
- USSTAB processing
- CLSDST pass VTAM application program
- USERVAR processing
- Use of VTAM generic resources

The first four choices are available to users of IMS/ESA Version 5 or 6. The last choice is only available to users of IMS/ESA Version 6.

There are other factors that might influence workload balancing. These include the use of session managers and TCP/IP connections. They are discussed in 9.2.8, “Network Workload Balancing With Session Managers” on page 76 and 9.2.9, “Network Workload Balancing With TCP/IP and TN3270” on page 79.

To facilitate a discussion of the alternatives, the following example is used.

Assume the Parallel Sysplex environment consists of two IMS systems. If one wished that 50 percent of the workload would be processed on IMS1 and 50 percent of the workload would be processed on IMS2, one could approximate that workload distribution by having 50 percent of the end users log on to IMS1 and the remaining 50 percent of the end users log on to IMS2. If this 50/50 split of the network workload can be achieved, an application workload split of 50/50 is also a likely result as long as IMS1 and IMS2 are clones of one another.

### 9.2.1 Instructions to the End-Users

The easiest way, from a technical perspective, that network workload balancing can be achieved is to tell 50 percent of the end users to log on to the APPLID of IMS1 and the other 50 percent to log on to the APPLID of IMS2. However, this is a tough sell, since most installations do not want to change end-user procedures every time an IMS system is added to the IMSplex, nor is there any way to guarantee that the end-users would log on as instructed.

### 9.2.2 USSTAB Processing

If end-users used VTAM USSTAB processing to log on to IMS, the VTAM definitions can be changed so that 50 percent of the VTAM node definitions pointed to one USSTAB table (for IMS1) and 50 percent of the VTAM node definitions pointed to another USSTAB table (for IMS2). However, this technique is very labor intensive to set up and maintain (massive changes when the load balancing objectives change and continuing maintenance for new terminals). In addition, this is likely to be a tough sell to the VTAM system programmers (the people that would have to do the work).

This brings us to the other three ways of accomplishing network workload balancing. These are the inclusion of a VTAM CLSDST PASS application program, VTAM USERVAR processing, and the use of IMS′s support of VTAM generic resources.

### 9.2.3 CLSDST PASS VTAM Application Program

The concept of a CLSDST PASS VTAM application program is as follows:

- The-end user logs on to a VTAM application program.
- The VTAM application program optionally requests information from the end-user, such as User ID.
- The VTAM application program selects a target IMS system (IMS1 or IMS2) based on hashing the end-user node name or other information provided by the end user.
- The VTAM application program transfers the end-user to the desired IMS system by issuing a VTAM CLSDST PASS request.

The primary advantages of this technique of network balancing are:

- No VTAM user exits are involved.
- The VTAM application program can issue other VTAM commands to determine resource (target IMS systems) availability.
- If desired, information other than the end-user′s node name can be used to determine the target IMS system.

- If desired, the VTAM application program can participate as a member of a VTAM generic resource group, thus allowing multiple copies of the application program to be active at the same time for availability.

The primary disadvantage of this technique of network balancing is that the end-user node will see the unbind and rebind SNA flows associated with VTAM CLSDST PASS processing. This should not be a problem for normal terminals, but could cause some problems for programmable terminals such as CICS/ESA FEPI (Front End Programming Interface) applications or workstation programs.

The use of a VTAM CLSDST PASS application program should be considered if one is using IMS/ESA Version 5. With IMS/ESA Version 6, direct support of VTAM generic resources by IMS eliminates the need for this solution in most installations.

### 9.2.4 USERVAR Processing

VTAM USERVAR processing was invented for IMS and CICS XRF (eXtended Recovery Facility), but has uses other than for XRF. One can think of USERVAR processing as indirect addressing for APPLID names. Rather than specifying an APPLID in a USSTAB or logon request, one can specify the name of a USERVAR. The contents of the USERVAR are then used by VTAM to obtain the actual APPLID name (in effect, VTAM uses the contents of the USERVAR to transform the USERVAR name to an APPLID name).

The key to using USERVAR processing to balance logon requests is to control how the USERVAR transformation process occurs. If one can ensure that 50 percent of the logon USERVAR transformation operations resulted in an APPLID of IMS1 and that 50 percent of the transformations resulted in an APPLID of IMS2, the objective of a 50/50 split of logon requests will be met.

There are two ways to make this happen.

The contents of a USERVAR can be changed through a VTAM command. Therefore, one can provide automation to periodically issue VTAM commands to change the contents of the USERVAR such that it contained IMS1 50 percent of the time and IMS2 50 percent of the time. The advantages of this technique are that it does not require any VTAM exits, and the automation product or tool can look at the current logon distribution and adjust its algorithm accordingly. The disadvantages of this technique are that there is no repeatability for a given LU to be logged on to the same IMS system each time it logs on, clustered logons (logons occurring within the same, relatively short time period) would tend to be routed to the same IMS system, and automation would probably incur more overhead.

The second technique is to use a VTAM exit called the USERVAR exit. If a USERVAR exit is provided to VTAM, it is called each time VTAM is requested to perform a USERVAR transformation. The USERVAR exit then performs the transformation. The USERVAR exit would probably perform some type of hashing or randomizing technique on the requesting node name in order to select the APPLID. For example, the USERVAR exit could add the first four characters of the node name to the last four characters of the node name, divide by 100, and select IMS1 if the remainder were 00 - 49 and select IMS2 if the remainder were 50 - 99. The advantages of the USERVAR exit technique are that the transformation for a given node name would be consistent, and there is low overhead. The disadvantages of this technique are that it requires writing a

VTAM exit in assembler (scares some people), and it assumes all IMS subsystems are available (the exit has no way of knowing if an IMS has failed).

It should be noted that the USERVAR repeatability advantage is lost if the end-user connects to IMS through a session manager. This is because most session managers maintain a pool of LUs for their use and the end-user is never guaranteed that the same LU from the pool will be used when the end-user logs on to IMS.

Whether automation were used or the USERVAR exit were used, one can create the USERVAR as a volatile USERVAR. If the adjacent SSCP facilities of VTAM are used, the automation or USERVAR exit would only need to run in one of the VTAM domains within the network.

The use of USERVAR processing should be considered if one is using IMS/ESA Version 5. With IMS/ESA Version 6, direct support of VTAM generic resources by IMS eliminates the need for this solution.

For additional information on VTAM USERVAR processing and the VTAM USERVAR exit, see the *Resource Definition Reference*, *Customization*, and *Operation* manuals for your VTAM release.

A sample USERVAR exit for network balancing is included in Appendix C, “Sample USERVAR Exit for Network Balancing” on page 219.

### 9.2.5 Use of VTAM Generic Resources

IMS/ESA Version 6 implements support of VTAM generic resources. This support allows end-users to log on using a generic name rather than with a specific IMS APPLID when requesting a session with one of the IMSs in a generic resource group. VTAM routes the logon request to one of the IMSs in the generic resource group based upon the following algorithm:

- If the terminal has an existing affinity for a particular IMS, the session request is routed to that IMS.
- In the absence of an existing affinity, VTAM:
  - Selects an IMS based on existing session counts. VTAM attempts to equalize the session counts among the generic resource group members.
  - Accepts a specific IMS as recommended by the MVS workload manager. The MVS workload manager must be in goal mode.
  - Invokes the user′s VTAM Generic Resource Resolution Exit, if present. This exit can select any of the IMSs in the generic resource group as the recipient of the session request.

The preceding algorithm does allow for three exceptions. First, for locally attached LUs, VTAM applies the algorithm to the IMSs, if any, that are executing on the MVS to which the LUs are attached. A ′local′ IMS in the generic resource group will be selected if at least one is active.

Secondly, end-user requests for a session with a specific IMS are honored and bypass the algorithm entirely. Session counts for local terminals and those established through requests for a specific IMS are taken into account in terms of VTAM′s attempt to balance the network workload by spreading the sessions evenly across all of the IMSs in a generic resource group.

Lastly, VTAM attempts to equalize session counts across all of the IMS subsystems in the generic resource group without regard for other differentiating factors, such as the fact that the various IMS subsystems may be executing on processors with different capabilities. Using the MVS Workload Manager in goal mode is a way to try and level these sorts of differences. The Workload Manager selects an IMS based on which IMS in the generic resource group is most easily meeting its goal. The VTAM Generic Resource Resolution Exit may be implemented to target session requests among the IMSs based on whatever algorithm it chooses. A possible advantage of using the Generic Resource Resolution Exit is the assignment of a session request for a given node name to the same IMS can always be guaranteed as long as that specific IMS is active. Use of the exit in this manner can be beneficial from a performance and/or usability point of view.

### 9.2.5.1 Affinities

An affinity for a particular IMS is established when a session is established with that IMS. IMS allows two options when requesting the use of VTAM generic resources. The option selected directly affects whether affinities are held or released when sessions are terminated, normally or abnormally.

### 9.2.5.2 VTAM Generic Resource Execution Option

An IMS execution option, GRAFFIN=IMS | VTAM, allows the IMS user of generic resources to specify whether (a) IMS or (b) VTAM is to manage affinity deletion. The second option, allowing VTAM to manage affinity deletion, is delivered by APAR PQ18590. Given the two choices, it is best to discuss them separately because the selection of one versus the other has a direct bearing on the availability of the IMSs in the Parallel Sysplex to the terminals in the network.

### 9.2.5.3 Use of VTAM Generic Resources: IMS Manages Affinity Deletion

Affinities are deleted during normal or abnormal session termination unless one or more of the following is true:

- The terminal retains significant status. Significant status is discussed beginning with the paragraph following this list.
- Session termination is for a SLUTYPEP or FINANCE terminal.
- Session termination is for the last ISC parallel session with a given node and not all parallel sessions, including this last session, to that node have been quiesced.

Significant status is set for a terminal if it is in response or conversational mode of execution or when the terminal is in some preset state or mode of operation, such as MFSTEST, test, preset, or exclusive. Significant status is always maintained for a static terminal whether its session is active or not; significant status is maintained for a dynamic (ETO) terminal when the user is signed on and is reset when the user is signed off (normally or abnormally).

The type of system failure, IMS or MVS, also determines whether affinities are released when the failure occurs. When the failure is an IMS failure, IMS′s ESTAE processing releases all affinities unless the terminal has significant status, is a SLUTYPEP or FINANCE device, or is associated with an ISC connection where all parallel sessions have not been quiesced.

If sessions are lost because of an MVS failure, IMS′s ESTAE routine does not get control, and affinities are not deleted regardless of whether significant status is held. Two courses of action are available for session reestablishment when sessions are terminated because of an MVS failure. First, the end user can log on using the specific IMS APPLID of one of the surviving IMSs. Or, the end user must wait until the failed IMS is restarted and then log on with the generic resource IMS name. This IMS restart can occur on any of the surviving MVSs in the Parallel Sysplex. SLUTYPEP and FINANCE terminals, most likely, will not be able to request a session using a specific IMS APPLID if they are already programmed to request a session using a generic resource name.

Dynamic terminal users will not always be handled consistently when sessions are terminated because affinities for non-STSN dynamic terminals are always deleted when a session is terminated unless they are terminated by an MVS failure. For example, a dynamic terminal user in conversational mode of execution at the time of session failure is placed back in conversational mode if the logon request, subsequent to the failure is passed to the IMS that owns the conversation. It will not be placed in conversational mode if the logon request is passed to some other IMS in the generic resource group. The Signoff and Logoff exits in IMS/ESA Version 6 have been changed in an attempt to address the inconsistencies that ETO terminal users might encounter.

The Signoff and Logoff exits in IMS/ESA Version 6 are now allowed to reset significant status whenever a session is terminated unless it is terminated by an MVS failure. This new capability of these two exits, if used, will help to minimize the inconsistencies a dynamic terminal user can see whenever a logon is processed after an IMS or session failure. Nevertheless, inconsistencies might still occur if the session failure is caused by an MVS failure.

In the event of an IMS failure, users of SLUTYPEP or FINANCE terminals and ISC sessions must wait until IMS is restarted before they can log back on and resume processing. This is not a change from the non-Parallel Sysplex world. Nevertheless, session availability for these terminal types, most commonly SLUTYPEP and FINANCE, can be of great importance.

The advantage of this implementation of VTAM generic resources by IMS is that the terminal, because affinities are retained if significant status is held, will be returned to the state it was in prior to session termination.

#### 9.2.5.4 Use of VTAM Generic Resources: VTAM Manages Affinity Deletion

One can choose to allow VTAM to manage affinity deletion. This option results in:

- The deletion of most affinities whenever a session is terminated (normally or abnormally). The phrase ′most affinities′ refers to ISC sessions as an exception. IMS continues to manage affinities for ISC sessions as described previously.
- Significant status is automatically reset at signoff, logoff, and IMS restart. An exception to this statement is the resetting of Fast Path response mode. Fast Path response mode can only be reset by an IMS cold start or by a COLDCOMM restart.

When VTAM is selected to manage affinity deletion, some of the considerations change in contrast to the previous discussion when IMS is selected to manage affinity deletion:

- SLUTYPEP and FINANCE terminal affinities, significant status, and sequence numbers are reset upon session termination. These terminal types are free to immediately log back on using the generic resource name no matter what the cause of the session termination. Session reestablishment, however, will be cold. The IMS subsystem to which they were previously attached has discarded the sequence numbers required for a warm start, and certainly, any other IMS with which a session may be established has no sequence numbers.

  The good news for SLUTYPEP and FINANCE terminals is that they can immediately attempt to reestablish a session with any member of the generic resource group. They do not have to wait for a failed IMS to be restarted when an IMS or MVS system failure is the cause of the session outage. The point that must be considered is that message integrity might be compromised because the sessions are always started without any attempt at resynchronization (Resynchronization is not possible because the required sequence numbers are never available.). Can the exposure of message integrity be tolerated? This can only be answered by the personnel knowledgeable in the applications involved, both on the host running under IMS and on the outboard SLUTYPEP or FINANCE terminals.

- Dynamic (ETO) terminal session reestablishment is handled consistently. Since significant status is always reset for all terminals, whether dynamic or static, STSN or non-STSN, a subsequent session establishment, after a failure, will always be accomplished without significant status as a consideration regardless of which IMS is selected as the session partner.

- Resetting of significant status includes exiting any active conversation whenever a session is terminated. Once again, whether this action is tolerable can only be decided on a system-by-system basis.

### 9.2.6 VTAM Generic Resources for APPC/IMS

LU 6.2 nodes can use VTAM generic resources with IMS. When an LU 6.2 node requests a session, it can specify a generic name, not a specific APPC/IMS ACBNAME. For example, if two IMS systems have APPC ACBNAMES of LU62IMS1 and LU62IMS2, they might have a generic name of LU62IMS. The LU 6.2 session request would be for LU62IMS. VTAM generic resources would select either LU62IMS1 or LU62IMS2 for the partner node. Since APPC/IMS does not use the same ACB as used by other IMS SNA nodes, it does not use the same generic resource group name. Instead, it must use a name associated with APPC/IMS ACBs. The support for generic names with APPC/IMS is provided by APPC/MVS. This support requires OS/390 Release 3 and VTAM 4.4 or higher releases.

The name for a generic resource group used by APPC/IMS is specified in the GRNAME parameter of the LUADD statement in the APPCPMxx member of SYS1.PARMLIB. If the GRNAME parameter is specified for an APPC/IMS LU, it will join the specified group when it is activated. All APPC/IMS instances belonging to the same group must specify the same group name. This group name must differ from the group name used by IMS for other LU types.

## 9.2.7 Printers and Network Workload Balancing

Printers can require special consideration in a multiple IMS Parallel Sysplex environment. It is likely that output to a particular printer can be generated by any IMS in the IMSplex, particularly if IMS systems are cloned. One has the option of sharing or not sharing printers among all of the IMSs.

If printers are shared (OPTIONS=SHARE for static terminals or the use of AUTOLOGON with dynamic terminals), multiple printer logons and logoffs will occur if multiple IMS subsystems are generating output for the printers. If this logon/logoff activity is excessive, print capacity can be greatly degraded and might result in excessive queuing (message queue data sets or shared queue structures filling up.)

One certainly has the option to not share printers. That is, a given printer or set of printers can be owned by one of the IMSs within, or outside, the IMSplex. Output for these printers from the non-printer owning IMS subsystems can be delivered to the printer-owning IMS:

- Through MSC sessions when the printer-owning IMS is not a member of the shared queues group that is generating the printer output or the IMS systems generating printer output are using traditional queuing. When MSC is used, the printer-owning IMS may or may not reside in the IMSplex (that is, within the same Parallel Sysplex environment). If MSC is to be used, the printer can not be a member of the shared queues group that is generating printer output.
- Through shared queues where the printer-owning IMS and all of the IMSs generating output for the printers are in the same shared queues group.

### 9.2.7.1 Printer Acquisition

A brief discussion of the different ways printer sessions with IMS can be established is appropriate at this point. One should understand the implications of printer usage upon how the sessions are established.

Printer sessions with IMS are typically established in one of three ways: by using IMS′s AUTOLOGON and/or printer sharing capability, by issuing an /OPNDST command, or by specifying LOGAPPL=applid on the printer LU definitions to VTAM.

The use of the /OPNDST command is likely to be most usable if a ′printer IMS′ is used. Automation can be used to issue the command whenever the printer IMS is started. Also, keep in mind, the /OPNDST command has a ′Q′ option which queues session requests to VTAM if the printer cannot be acquired immediately. The /OPNDST command, itself, can be either manually entered or submitted from an automated operator program or through some other means of automation. Operator intervention is likely to be required when printer sessions are lost.

Printer sharing or AUTOLOGON causes IMS to automatically acquire a session with a printer whenever IMS queues output to a printer. One must be careful, when either or both of these techniques is used, if the printers are to be shared among the members of a shared queues group because of the possible loss of print capacity that might result from repeated establishment and reestablishment of printer sessions.

The use of printer sharing or AUTOLOGON by a ′printer IMS′ does not result in a loss of print capacity because the printers are only in session with a single IMS.

If a ′printer IMS′ is to be used, automatic session initiation only occurs when the output is generated by the ′printer IMS.′ Output generated by the other IMSs in the share queues group is simply queued on the shared queues. The printer IMS does not register interest in a printer LTERM queue until output for a given printer LTERM is queued by the ′printer IMS,′ thus causing the printer IMS to acquire a session with that given printer. This consideration or concern with the use of a printer IMS is likely to be minimized if systems are cloned, or might be greatly magnified if the IMS systems in the shared queues group are partitioned by application and printer usage is application related.

Defining printers to automatically establish sessions with IMS by specifying LOGAPPL=imsapplid, whenever both the printer and IMS are active, is an acceptable way to establish sessions. The use of LOGAPPL= does tie a printer to a given IMS. The LOGAPPL= specification can be overridden by operator action (can even be automated using a product such as NetView). This might be necessary if the IMS which is specified in the LOGAPPL= operand is not available. Of course, if LOGAPPL= is used and the specified IMS is unavailable, one can choose to always wait for the unavailable IMS to become available.

IMS′s VTAM generic resource name can be specified as the LOGAPPL= keyword option. If a printer IMS is to be used, then a VTAM Generic Resource Resolution Exit is required to route the printer session requests to the printer IMSapplid. The use of IMS′s generic resource name as the LOGAPPL= specification gives us an opportunity to dynamically switch the ′printer IMS′ from one APPLID to another in the event the current ′printer IMS′ is unavailable for whatever reason. The Generic Resource Resolution Exit input interface includes a list of all active generic resource group members.

### 9.2.7.2 Printer Queue Buildup

Despite the best laid plans, whether full-function traditional or shared queues are in use, message queue data sets or shared queue structures can become full. When message queue data sets are used, this queue-full condition results in an abend (U0758) of the entire IMS subsystem. With shared queues, an abend (U0758) of an IMS subsystem is less likely to occur, but the disruption caused by the queue structure full condition is likely to be noticeable, particularly if the queue structures are full of printer output that cannot be delivered (for instance, the ′printer IMS′ is not available).

With shared queues and a structure-full condition, PUTs to the shared queues are rejected. The rejected message is either discarded or temporarily stored in IMS′s queue buffers. If an IMS runs out of queue buffers, that IMS subsystem is aborted with a U0758 abend. Whether full-function traditional or shared queues are used, a queue-full condition is something to be avoided.

What does a message queue data set IMS user do to avoid a queue-full condition? They define overly large message queue data sets and take action when queue data set usage thresholds are reached.

What can a shared queue IMS user do to avoid a queue-full condition? If the fear is they might become full because of queuing large volumes of printer output (which is the subject of this section of the manual), one solution is to remove a ′printer IMS′ and the printer output queues from the shared queues group. With this solution, printer output is still generated by the members of the shared queues group and placed on the shared queues.

One or more members (more members reduces the possibility of failure) of the shared queues group establish MSC sessions with the ′printer IMS.′ All printer output is destined for the ′printer IMS′ which is remote to the shared queues group. As printer output is queued to the remote ready queue, it is immediately sent across an MSC link to the ′printer IMS′ outside the shared queues group. In this fashion, the residency of the printer output on the shared queues tends to be transitory; that is, there should be no build-up of printer output on the shared queues structures unless the ′printer IMS′ is unavailable.

Given that the ′printer IMS′ is in a Parallel Sysplex environment, the use of ARM (Automatic Restart Manager) with the ′printer IMS′ will minimize any outage of a ′printer IMS.′ The ability of ARM to restart a failed printer IMS in a timely manner might be deemed to be a sufficient solution to the problem of a ′printer IMS′ outage.

### 9.2.7.3 Considerations for Sharing Printer Workload Among Multiple IMSs

Listed below are several comments that apply when the printer workload is shared among the IMSs in a shared queues group:

- System definitions among the IMSs can be cloned.
- The use of AUTOLOGON and/or IMS-supported static printer sharing are likely to be the most efficient techniques used to acquire printer sessions.
  - Multiple IMSs might compete to send output to the same printer if the systems are cloned. This can adversely affect printer capacity to the extent the shared queues might overflow and even become full (out of space or structure storage).
  - If the IMS systems are partitioned (not cloned), the competition for the printers among the IMSs in the shared queues group is most likely to be less than if the systems were cloned. The degree to which this statement is true is dependent upon the extent to which the printers are shared among the partitioned applications and upon whether these applications execute on the same or separate IMSs.
- In the event of an IMS failure, the ability of the surviving IMSs in the shared queues group to deliver output to the printers is not impaired.

If a printer-owning IMS solution is chosen, it is important that the IMSplex implementation include procedures for actions to be taken if the printer-owning IMS system fails. Some food for thought follows:

- Our recommendation is to wait for the printer-owning IMS to be restarted and resume its print responsibilities. The ′printer IMS′ should be in a Parallel Sysplex environment where it can take advantage of the Automatic Restart Manager (ARM) for quickly restarting the ′printer IMS.′ This choice is practical from an implementation and operational point of view unless the printer-owning IMS cannot be restarted in a timely fashion. The fact that the printer IMS cannot be restarted by ARM in a timely fashion can be considered to be a double failure.
- The only alternative to the preceding discussion is to shift the print responsibility to one of the surviving IMSs:
  - In a non-shared queues Parallel Sysplex environment, this requires the use of back-up MSC links to physically reroute the printer traffic. The `/MSASSIGN` command can be entered on each of the surviving IMSs that

have MSC connections to the new ′printer IMS′ to allow the printer traffic to be routed to it.

- With all IMSs as members of a shared queues group, the failure of the ′printer-IMS′ requires that one of the surviving IMSs be selected to become the printer-IMS. This could be accomplished by simply establishing sessions with the printers by issuing /OPNDST commands on the new ′printer IMS.′ Alternatively, if LOGAPPL=imsgenericresource name, the VTAM Generic Resource Resolution Exit can be notified (for example, through the use of the VTAM MODIFY EXIT,PARMS= command) to route session requests to an alternative ′printer′ IMS.

The preceding discussion on handling printers might have given the impression that the use of a ′printer IMS′ is recommended. This is not true. A ′printer IMS′ is only recommended for use if printer sharing among the IMSs producing printer output cannot be delivered in a timely fashion such that the queue data sets and/or queue structures are in danger of becoming or become full. Otherwise, there is nothing wrong with sharing the printers among all of the IMSs producing printer output.

## 9.2.8 Network Workload Balancing With Session Managers

Many installations use session managers to handle one or more VTAM sessions for an end-user. Session managers are VTAM applications that create sessions with other VTAM applications, such as IMS. In a typical use, an end-user logs on to the session manager. Then, the session manager creates multiple sessions for the end-user. They could be with one or more IMS systems, CICS systems, TSO, or other applications.

The use of a session manager somewhat complicates workload balancing. It can occur in either of two places. If there are multiple instances of the session manager, the balancing can occur by spreading the work among session managers. In this case, each session manager can not have to balance work among multiple IMS systems in the Parallel Sysplex. If there is only one instance of the session manager, workload balancing must be done when the session manager creates sessions with IMSs in the Parallel Sysplex. We will look at both of these cases.

### 9.2.8.1 Using One Instance of the Session Manager

IMS/ESA Version 6 can use generic resources with session managers. If there is only one instance of the session manager, its placement might affect session balancing. If an application that is part of a local SNA major node initiates a session to a generic resource, VTAM first attempts to establish a session with a generic resource member on the same VTAM node. That is, the session manager will establish a session with the IMS on the same MVS. This would eliminate balancing of sessions from the session manager. All sessions would be established with the IMS on the same MVS with the session manager. VTAM provides a generic resource resolution exit that can be used to change this selection. The generic resource resolution exit (ISTEXCGR) can override the selection of a generic resource member on the same node. Instead, the usual selection criteria can be used to balance the sessions across all generic resource members. This capability of the exit routine was added by VTAM APAR OW30001.

The following example illustrates a possible use of one instance of the session manager. Generic resources are used by IMS, but the Generic Resource

Resolution Exit is not used. In this example, the session manager resides on MVS1. IMS1 is on MVS1. IMS2 is on MVS2, and IMS3 is on MVS3. An attempt to log on to IMS from the session manager will always result in a session with IMS1. This will eliminate any balancing. Without the exit, VTAM will select an IMS instance on the same node (MVS) with the session manager when one exists. This is illustrated in Figure 4.

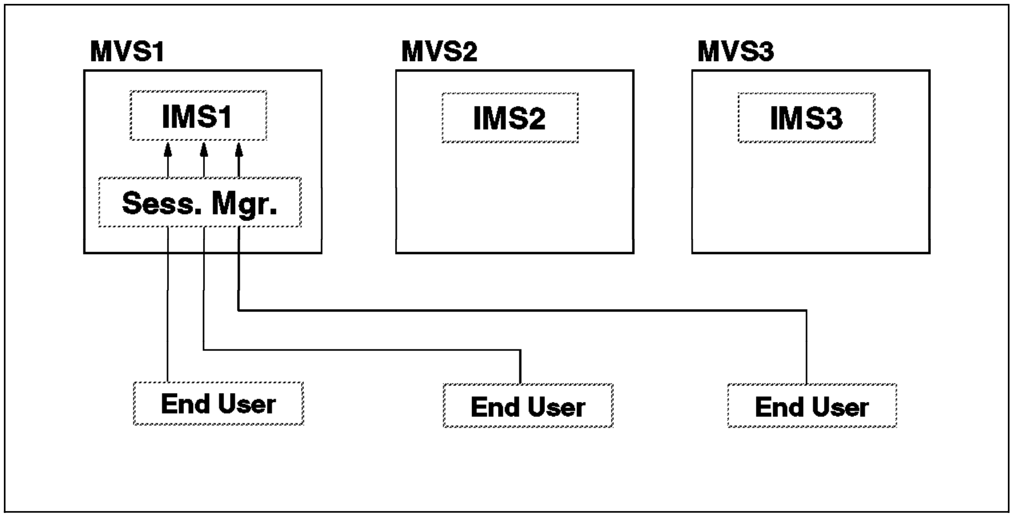


Figure 4. Using One Instance of a Session Manager Without ISTEXCGR

The VTAM generic resources exit (ISTEXCGR) can be used to balance the sessions. A possible result of the use of this capability is shown in Figure 5.

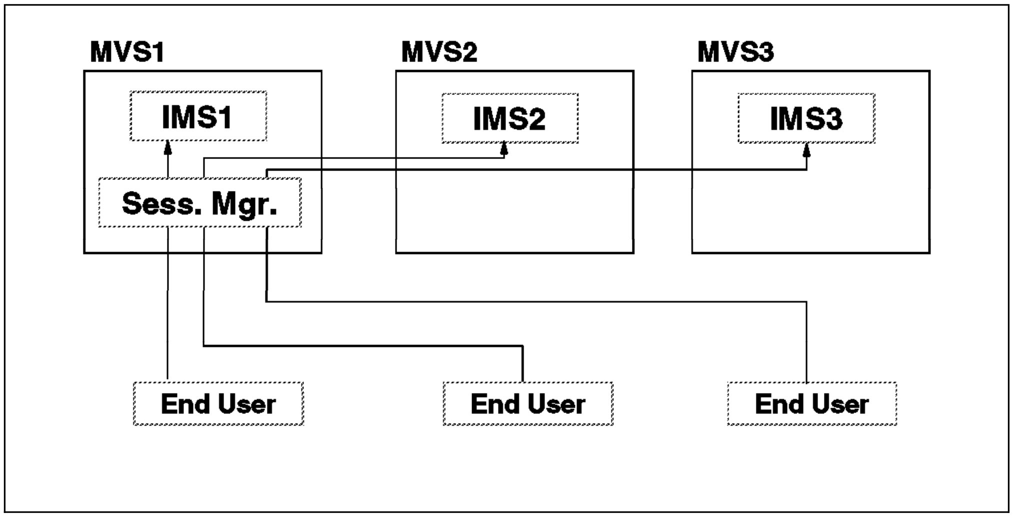


Figure 5. Using One Instance of a Session Manager With ISTEXCGR

There is another way to balance sessions with IMS when using only one instance of the session manager. This is to place the session manager on an MVS which does not contain an IMS instance in the generic resource group. Since there is no local IMS, the ISTEXCGR exit routine does not have to be used to balance sessions. The normal VTAM generic resource processes will balance the sessions. This is illustrated in Figure 6 on page 78.

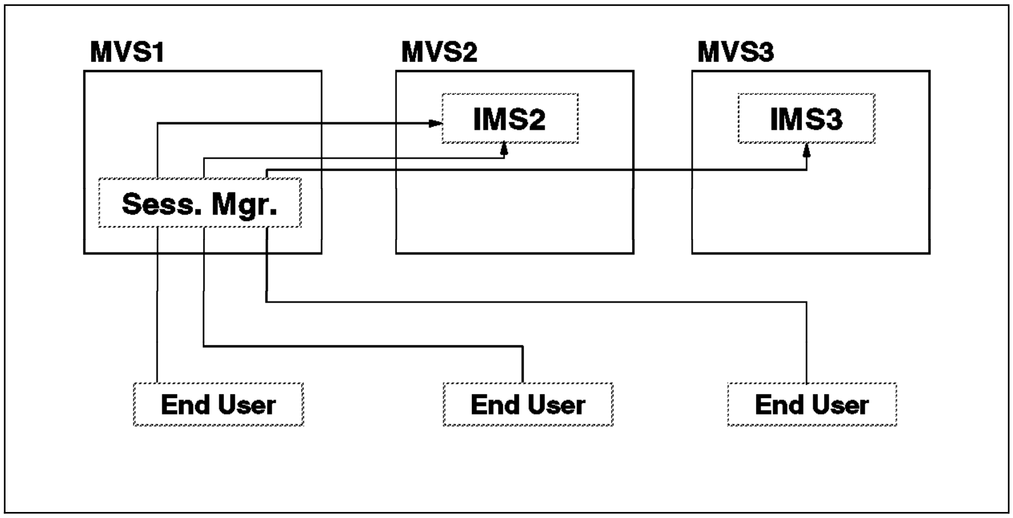


Figure 6. Placing the Session Manager on an MVS Without IMS

### 9.2.8.2 Using Multiple Instances of the Session Manager

Some installations might want to have multiple instances of their session manager. This provides availability benefits as well as additional options for workload balancing.

If the session manager supports generic resources, the logons to the session manager can be balanced. If each MVS system has both a session manager instance and an IMS instance, the balancing of the logons across the session managers allows one to always use the local IMS when establishing sessions between the session manager and IMS. This is illustrated in Figure 7.

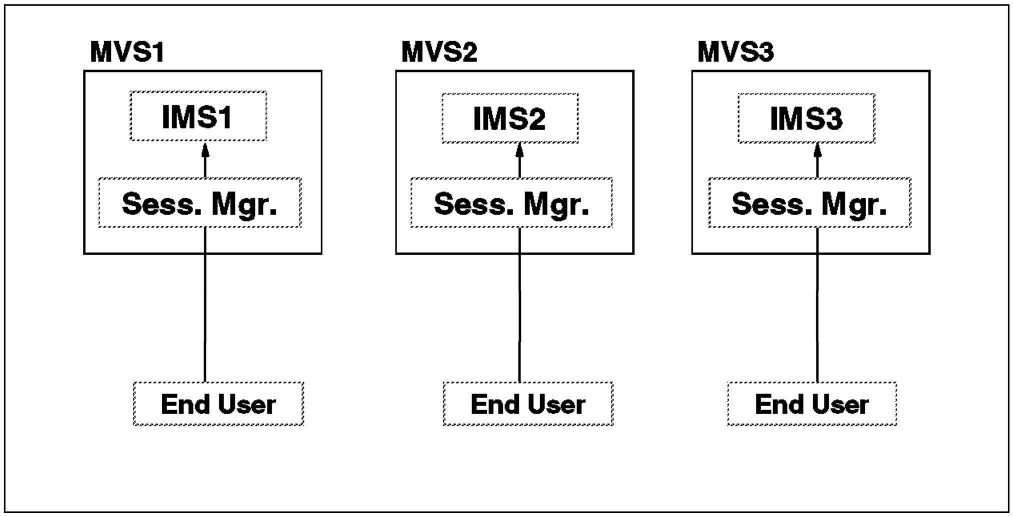


Figure 7. Session Managers With Generic Resource Support

Of course, there are many other possible configurations for sessions managers and IMS instances.

Some installations will receive the greatest balancing and availability benefits by using generic resources with both IMS and the session managers. This will tend to balance the use of session managers and IMS systems, even when they are not paired on each MVS system.

Some session managers do not have generic resource support. Installations with these session managers give their end-users instructions about which session managers are available to them. Such installations can use generic resources with IMS to balance the workload across the IMS systems.

## 9.2.9 Network Workload Balancing With TCP/IP and TN3270

Users of TCP/IP can take advantage of workload balancing capabilities provided by Domain Name Server (DNS) in OS/390 Version 2 Release 5 and later releases. They are also available in Release 4 through a no-charge kit. These DNS capabilities are appropriate for users with long-term connections, such as TN3270 users.

DNS allows a user to resolve a name request to one of multiple IP addresses in a Parallel Sysplex. Each connection request is assigned an actual IP address based either on a weight assigned to each IP address or on Workload Manager capacity and workload information. DNS provides functional equivalence with VTAM generic resources for TCP/IP users, such as TN3270. As is explained below, TN3270 users can use both DNS and generic resources.

### 9.2.9.1 DNS Without Generic Resources

The following example illustrates a possible use of this DNS function. Assume that IMS1 is executing on MVS1, IMS2 on MVS2, and IMS3 on MVS3. Each MVS system has an instance of TN3270. A TN3270 client logs in to TN3270. DNS resolves the request to one of the TN3270 instances. This balances TN3270 login requests across the instances of TN3270 servers. The user then logs on to IMS establishing a VTAM session between the TN3270 instance and an IMS. If VTAM generic resources are not used, the user must either log on to a specific IMS or use a USERVAR exit to select a particular IMS. If the user must choose a specific IMS instance, DNS does not provide any workload balancing. USERVAR exits can be used to provide workload balancing. In this scheme, each MVS would have a different USERVAR exit. Each exit would assign a value equal to the IMS instance executing on its MVS. This scheme is illustrated in Figure 8. It shows USERVAR exit routines which always assign an IMS log on request to the IMS on the same MVS.

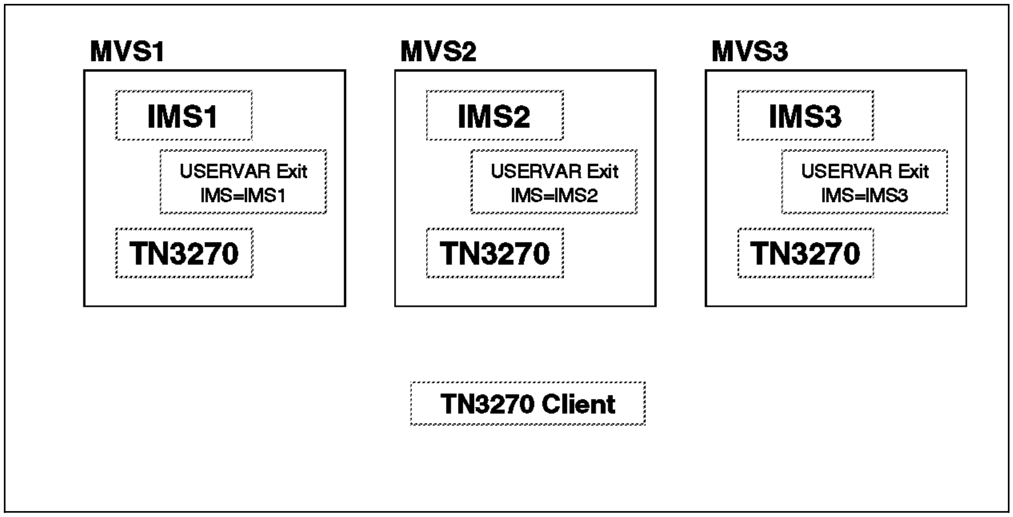


*Figure 8. Using DNS With TN3270 and VTAM GR*

This scheme could be enhanced to handle IMS failures. If an IMS system fails, the USERVAR exit routine could be dynamically modified to route new logon requests to an IMS on another MVS.

### 9.2.9.2 DNS With Generic Resources

VTAM generic resources can be used in conjunction with TN3270 and DNS. The following example illustrates a possible use of this combination. It is similar to the example in Figure 8 on page 79 but USERVAR is not used. Instead, GR is used. The log on requests to IMS from TN3270 will always result in a session with the IMS on the same MVS. This occurs because TN3270 and the IMS reside on the same node. GR will always select a local generic resource if one is available unless the VTAM generic resources exit (ISTEXCGR) indicates otherwise. This example is illustrated Figure 9. It shows the TN3270 login request resulting in a connection to the TN3270 on MVS1. The IMS logon request results in a connection to IMS1.

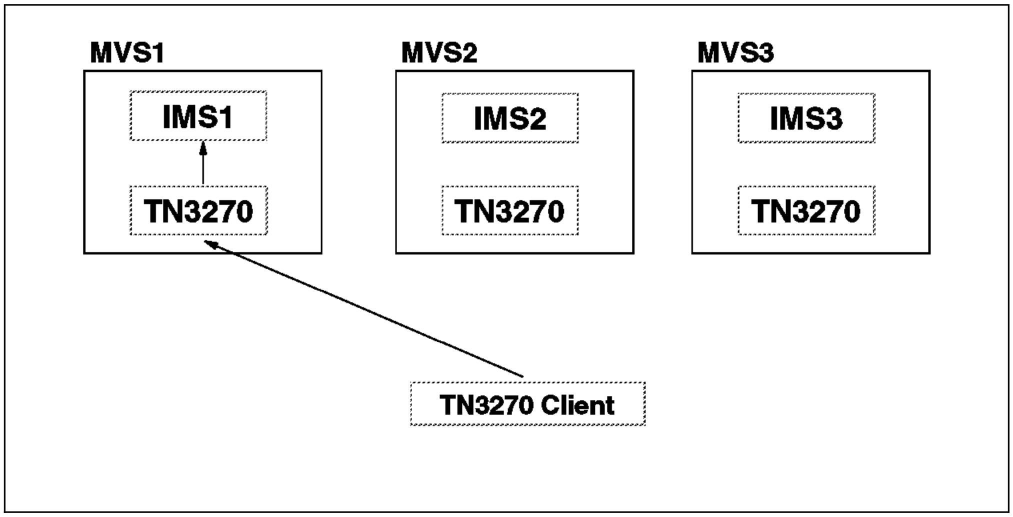


*Figure 9. Using DNS With TN3270 and VTAM GR*

If the IMS on the same MVS with the TN3270 being used is not available, the VTAM logon request will result in a session with one of the IMS instances on another MVS. The IMS instance chosen depends on the usual VTAM generic resources considerations. This example is illustrated by Figure 10 on page 81. It shows the IMS logon request resulting in a session with IMS2.

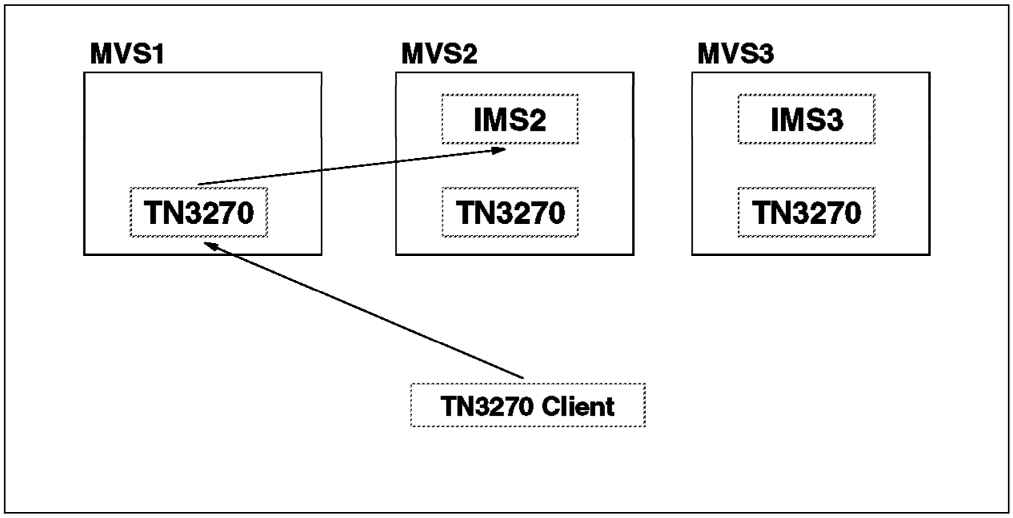


*Figure 10. Using DNS With TN3270 and VTAM GR After a Failure of IMS1*

## 9.3 Application Workload Balancing

Application workload balancing attempts to balance the transaction workload among the IMS systems in the IMSplex. Depending upon how network balancing is achieved, your IMSplex configuration (cloned, partitioned, and so on.), and your workload balancing objectives, application workload balancing may or may not be an automatic result. If application workload balancing objectives are not achieved, additional implementation choices must be considered and selected to ensure the resulting application workload distribution satisfies your objectives.

There are several implementation choices to achieve application workload balancing, including:

- Network workload balancing
- MSC balancing
- IMS Workload Router
- Shared Queues

The first three choices are available to users of IMS/ESA Version 5 or 6. The last choice is only available to users of IMS/ESA Version 6.

### 9.3.1 Network Workload Balancing

The various techniques to accomplish network workload balancing were discussed in the previous section, 9.2, “Network Workload Balancing” on page 66.

If one wished that 50 percent of the application workload would be processed on IMS1 and 50 percent would be processed on IMS2, one could approximate that workload distribution by having 50 percent of the end-users log on to IMS1 and the remaining 50 percent of the end-users log on to IMS2. If this 50/50 split of the network workload can be achieved, an application workload split of 50/50 is also a likely result as long as IMS1 and IMS2 are clones of one another.

If systems are partitioned rather than cloned, network load balancing more than likely will not result in application workload balancing. A partitioned configuration by application, for example dictates where application processing will take place.

### 9.3.2 MSC Balancing

MSC balancing attempts to balance the workload among the IMS systems in the IMSplex by routing work to the various systems through MSC definitions and/or MSC exits. For example, one could have all of the end-users logged on to one of the IMS systems and have that system route some percentage of the transactions to another system or systems within the IMSplex. Let us assume, for example, that all of the end-users are logged on to IMS1. In addition, we want 40 percent of the transaction workload to process on IMS1 and the other 60 percent to process on IMS2. Given these assumptions, we would want IMS1 to route 60 percent of the transaction workload to IMS2.

This routing could either be based on specific transaction codes (transaction codes A through G process on IMS1 and transaction codes H - Z process on IMS2), or could be based on percentages (40 percent of transaction codes A - Z process on IMS1 and 60 percent of transaction codes A - Z process on IMS2). Routing based on specific transaction codes (is a possible solution when systems are partitioned by application) can be accomplished through the use of SYSGEN specifications. Routing based on percentages (is a possible solution when systems are cloned) requires the use of the MSC Input Message Routing Exit (DFSNPRT0), introduced in IMS/ESA Version 5.

The MSC Input Message Routing Exit, DFSNPRT0, has the capability to route transactions to a specific IMS system based on either MSNAME or SYSID. Thus, if one wanted 60 percent of all transactions entered on IMS1 (that′s where all the end-users are connected) to be routed to IMS2, one could write a DFSNPRT0 routine that would issue a STCK, divide the result by 100, and route the transaction to IMS2 if the remainder were 0 - 59.

### 9.3.3 IMS Workload Router

The easiest way to implement MSC balancing is through the *IMS/ESA Workload Router for MVS* (WLR), a separate product (5697-074).

The IMS Workload Router works with IMS TM to route or balance the IMS transaction workload across multiple IMS systems in a Sysplex or non-Sysplex environment. WLR provides flexibility in how the workload is routed and also allows routing to be changed dynamically.

WLR software is integrated with the IMS TM software and provides transaction workload balancing that is completely transparent to end-users and IMS applications.

WLR superimposes the roles of *router* (front-end) and *server* (back-end) systems on IMS systems in the WLR configuration. Routers route the transaction work load to servers; servers process the transactions and return any responses to the router. WLR supports a configuration of one or more router systems coupled with one or more server systems. This makes WLR adaptable to a wide variety of Parallel Sysplex configurations.

The use of WLR within a Parallel Sysplex data sharing environment to distribute application workload is inappropriate when shared queues are used. With shared queues, MSC links can be defined among the members of a shared queues group, but they cannot be used. IMS V6 tolerates the definitions, but does not allow MSC links between members of a shared queues group to be started. MSC links from one or more members of a shared queues group to IMS subsystems remote to the shared queues group are supported, and WLR can be employed to support these connections.

Highlights of WLR include the following:

- Using MSC, distributes IMS transactions that originate from network input or program-to-program message switches
- Provides for weighted distribution of transactions; that is, different server (back end) systems can receive different parts of the work load
- Provides an online, real-time administrative interface for monitoring and dynamically updating the WLR configuration
- Supports parallel MSC sessions between the router (front-end) and server (back-end) systems, thus facilitating MSC link balancing
- Automatically recognizes and avoids routing transactions to unavailable server systems and MSC links; automatically reconfigures the work load when planned or unplanned outages occur
- Has sophisticated surge detection and suppression capabilities to avoid potential performance problems
- Supports the notion of transaction “affinity;” assigns transactions or groups of transactions to a designated server system
- Is flexible and can be customized to meet the needs of different installations
- Offers server options for both cloned and non-cloned TM networks
- Requires no IMS user modifications to install

### 9.3.4 Shared Queues and Application Workload Balancing

First, the use of shared queues does impose a restriction. The use of active MSC links among the IMS members of a shared queues group is not allowed. The use of shared queues to distribute application workload is a replacement for distributing this workload using MSC. In reality, MSC sessions can not be established among the IMS systems in a shared queues group.

The use of shared queues introduces an entirely new set of application workload balancing considerations. With shared queues, IMS will process a full-function input transaction in the front-end if the following are true:

- A dependent region, with the ability to process the input transaction, is waiting for work.
- The input transaction fits into a single queue buffer logical record.
- The transaction is not defined as serial (SERIAL=YES specified on the TRANSACT macro).

The transaction is only placed on a shared queue and is eligible for IMSplex-wide processing if the preceding qualifications are not true. This bias for processing in the system where a transaction is entered is desirable in that it tends to minimize the overhead associated with the use of shared queues.

For EMH-driven transaction input, the Fast Path Input Routing Exit (DBFHAGU0) has three alternatives which affect or determine where an input transaction is processed. These alternatives are as follows:

- LOCAL ONLY: When specified, the input transaction is queued to the appropriate balancing group (BALG) in the front-end IMS system. These transactions are always processed in the front-end. No access to the shared queue EMH structure(s) on the coupling facility is required.

  The benefit of LOCAL ONLY is that it avoids all coupling facility overhead. On the other hand, LOCAL ONLY cannot take advantage of one key Parallel Sysplex advantage: the use of extra capacity for transaction processing by back-end IMSs in the same shared queues group when the transaction load exceeds the processing capacity of the front-end.

- LOCAL FIRST: When specified, an input transaction is queued to the appropriate BALG in the front-end if the queue count is five or less. If the queue count is greater than five, the input transaction is placed on the shared queue structure where it is available for processing by any IMS in the shared queues group eligible (any IMS with at least one IFP region active and sensitive to the appropriate PSB) to process it.

  The use of LOCAL FIRST favors processing of transaction input in the front-end unless queuing becomes ′excessive′. The definition of ′excessive′ has been defined arbitrarily as a BALG queue count greater than five transactions. LOCAL FIRST, then, attempts to minimize the overhead of shared queues by processing most input in the front-end, but will also be able to apply back-end processing capacity if the front-end should become overloaded.

- GLOBAL ONLY: When specified, GLOBAL ONLY causes all input transactions to be placed on the shared queues where they are eligible for Sysplex-wide processing. There is no bias to processing in the front-end.

  The use of GLOBAL ONLY incurs the maximum overhead of shared queues. That is, multiple accesses to the coupling facility shared queue structures per input transaction is guaranteed when GLOBAL ONLY is used. GLOBAL ONLY is only recommended when the shared queues configuration is made up of partitioned applications executing on a subset of the IMSs in the shared queues group or when the configuration is that of terminal-owning front-end IMSs feeding the transaction workload to transaction-processing IMS back-ends. Of course, LOCAL FIRST and GLOBAL ONLY are equivalent whenever the front-end system does not have a local IFP available to process an input transaction.

- LOCAL FIRST is the recommended choice. It combines the benefits of efficient processing in the front-end with the ability to take advantage of available back-end capacity when required. GLOBAL ONLY, of course, is a correct choice when the configuration (partitioned or front-end/back-end) requires it. We expect the use of LOCAL ONLY to be very limited because it is too restrictive, particularly when compared to LOCAL FIRST.

Given the preceding comments, the efficient use of **shared queues** favors a cloned environment where:

- All IMSs in the IMSplex can process all transaction input. This automatically solves application processing capacity problems. Another result of cloning is improved availability. If an IMS should fail, the surviving IMSs within the shared queues group can pick up the application processing workload.

- The network workload is spread evenly across all of the IMSs in the IMSplex. This minimizes the overhead of using shared queues and tends to spread the application workload evenly across all of the IMSs in the IMSplex.

While the efficient use of shared queues favors a cloned configuration, shared queues can be used to support a partitioned configuration. With shared queues, the absence of an active dependent region eligible to process a given transaction class results in the queuing of these transactions on the shared queue in the coupling facility. Then, all back-end IMS systems in the same shared queues group with interest in processing these transactions can retrieve them, process them, dequeue them, and place reply messages on the shared queues for the front-end IMS to send to the inputting terminal.

## 9.4 Transaction Scheduling Considerations

In a non-shared queues environment, the scheduling of a transaction is influenced by certain parameters specified on the TRANSACT macro. These parameters are related to queue length. With shared queues, IMS has no knowledge of how many messages are on any queue. The TRANSACT macro specifications of interest and their use with shared queues follow:

- PARLIM: Without shared queues, an additional region is scheduled whenever the current transaction enqueue count exceeds the PARLIM value multiplied by the number of regions currently scheduled for this transaction. With shared queues, because the queue count is not known, all PARLIM specifications are treated as if PARLIM=0 were coded. That is, any input transaction can cause a new region to be scheduled up to the MAXRGN= specification. The absence of a PARLIM specification still limits the scheduling of a transaction to a single, dependent region at a time within a given IMS member of a shared queues group.
- PRTY: This keyword option allows the priority of a transaction to be raised when the number of queued transactions is equal to or greater than the limit count value. With shared queues, both the LIMIT PRIORITY and the LIMIT COUNT values are ignored. In a shared queues environment, transactions are always scheduled according to the NORMAL priority definition.

## 9.5 Summary of Workload Balancing Considerations

This chapter separated workload balancing among the IMSs in a Parallel Sysplex environment into two types of workload: network workload and application workload.

There are various techniques and tools available to distribute the network workload. One is exclusive to IMS Version 6, and that is the use of VTAM′s generic resource function. As described, the generic resource function comes in two flavors: where IMS manages affinities or where VTAM manages affinities. The generic resource function may be used with or without shared queues.

Application (transaction processing) workload balancing may be a direct result of balancing the network workload. If the IMS systems are partitioned by application, the distribution of the application workload is predetermined. A front-end/back-end configuration pushes all of the application workload on the back-end systems where it may be shared among the back-end systems (the back-ends are clones of one another) or split among the back-end systems by isolating applications to a specific system or systems.

Several techniques for distributing application workload were discussed and included network workload balancing, the use of MSC, the IMS Workload Router, and shared queues.  Shared queues is a function that is only available with IMS/ESA V6.

To clone or not to clone applications across multiple IMSs within the IMSplex is a question that must be answered and incorporated in any migration plan.  The decision not to clone negates one of the advantages available with a Parallel Sysplex: An application outage cannot take full advantage, if any, of the processing capability of the surviving IMSs.

Lastly, users of shared queues have to understand that transaction scheduling does not consider queue counts when scheduling a dependent region.  Parallel transaction processing in dependent regions is controlled by the MAXRGN= specification on the TRANSACT macro without regard to queue buildup.  All transactions are scheduled based on NORMAL priority as specified on the TRANSACT macro.  LIMIT priority, if specified on the TRANSACT macro, is ignored.

# Chapter 10. IMS TM System Definition Considerations

**TM Only**

This chapter applies only to IMS TM.

This chapter discusses a number of system definition considerations for IMS subsystems in the IMSplex configuration. IMS/ESA Version 6 includes changes that enhance the ability to clone IMS subsystems. With IMS/ESA Version 5, special consideration had to be given to certain inhibitors to cloning that were removed in IMS/ESA Version 6.

## 10.1 Master Terminals With IMS/ESA Version 5

Each IMS TM subsystem is an independent system. As such, each IMS TM subsystem requires a set of master terminal definitions (primary and secondary), because the VTAM node names must be different in each system definition. In other words, if there are ′n′ IMS TM systems in the IMSplex, there will be ′n′ primary master terminals and ′n′ secondary master terminals.

There are two exception cases. If the VTAM nodes specified in the system definition do not exist (meaning not defined to VTAM), the master terminal node names specified in the IMSplex system definitions do not need to be unique. In this situation, the IMS TM systems will execute without a master terminal, or the master can be assigned to a different terminal in the system definition. Also, if the primary and secondary masters are BTAM 3270 local terminals, the same LINEGRP and TERMINAL macro definitions can be used in all of the system definitions as long as the DD statements created for the IMS PROC during stage 2 processing are adjusted to point to the appropriate, unique, local 3270 addresses.

Another item to consider is whether to use the same LTERM names for the master terminals on all of the systems. The choice of the same LTERM name or unique LTERM names has application considerations.

If application programs send messages to the master terminal (for example, unexpected status code messages, and so forth.) and you want these messages to appear on the master terminal for the system where the application program executes, the same LTERM name for the master terminals on all of the IMS TM systems in the IMSplex should be defined. However, if you want these application messages to be consolidated and sent to the same terminal, no matter on which system the application executed, unique LTERM names for the master terminals are required.

The LTERM name used for consolidated application messages (the master terminal LTERM name prior to cloning) can be a master LTERM name on one of the IMS systems, or can be a “new,” non-master LTERM. In any case, this LTERM is defined as a “local” LTERM on one of the IMS systems and as a “remote” LTERM on all of the other IMS systems in the IMSplex. All of the IMS systems within the IMSplex are then connected through intra-IMSplex MSC links which are used to route the application messages to the IMS system containing the terminal.

We recommend the use of unique LTERM names for the master terminals on all of the IMS systems in the IMSplex. This will make migration to shared queues easier. With shared queues, unique master terminal names of IMS systems in a shared queues group are required.

## 10.2 Master Terminals With IMS/ESA Version 6

IMS/ESA Version 6 removes the requirement to define unique master and secondary master terminals. The removal of this requirement is another step toward allowing cloned IMS definitions among IMSs in an IMSplex. This enhancement applies to shared queues and traditional queuing environments.

DFSDCxxx is a new IMS.PROCLIB member available with IMS/ESA Version 6 that allows one to:

- Override the primary and secondary master node names that have been generated by the system definition process at control region startup. As long as all systems utilize the same terminal type for the master and secondary master, systems within an IMSplex can be cloned from a master/secondary master perspective. Up to eight LTERM names can also be overridden for each master terminal, primary or secondary. Of the LTERM names provided by DFSDCxxx, the user can specify which overriding names are the MASTER and SECONDARY master LTERMs. The names specified in each DFSDCxx member (one member for each IMS in the IMSplex), of course, must be unique.

- Specify two generic LTERM names, one each for the primary and secondary master terminals. It is intended that the generic names be the same for all IMSs in the IMSplex. There are a couple of uses for generic primary and secondary LTERM names.

  When an application program inserts a message to the primary or secondary master generic LTERM name, the destination LTERM name is recognized by the IMS where the application is executing as the generic master or secondary master LTERM name and queues the message to the real master LTERM destination name of the IMS system where the insert was made. This capability allows us to clone our application programs that insert messages to the master or secondary master LTERMs without any program changes required.

  Similar results occur if terminal-to-terminal message switches are used to send messages to the master or secondary master LTERM. That is, if the terminal-to-terminal message switch destination name is a generic primary or secondary master LTERM name, the message will be queued to the real master or secondary master LTERM of the system on which the message switch was entered.

## 10.3 Security

When all of the IMS systems within the IMSplex are logically equivalent, they should all use the same security profile. In order for this to be implemented easily, all of the IMS systems within the IMSplex should be defined with the same value specified in the RCLASS parameter on the SECURITY macro. This allows all of the IMS systems to share the same RACF (or equivalent) security profiles.

In a shared queues environment, transactions can be processed in the front-end IMS (where the terminal user is signed on) or the back-end (where the terminal user is not signed on). When a transaction is processed in the back-end and security checking is required:

- The security environment is dynamically created (ACEE is built) in the back-end IMS when security checking is required. This is done using the user id that is in the message prefix of the input transaction.
- Any security check is accomplished without password verification (the password associated with the user id is not present in the message prefix). Security checks are made when an application program issues a call to a secured resource (CHNG, AUTH CMD and ICMD calls) and for deferred conversational program-to-program message switch.

Security-related IMS user exits are driven as appropriate. These exits are the Command Authorization (DFSCCMD0), Security Reverification (DFSCTSE0), and Transaction Authorization DFSCTRN0) exits. When driven in a back-end system, these exits are not passed the address of the CTB associated with the signed on user. The CTB is a resource owned by the front-end IMS in support of the terminal session and is not passed to a back-end system.

## 10.4 Resource Definitions and Usage

With shared queues, IMS does not ensure that resource definitions and usage are consistent among the members of a shared queues group. For example:

- The same resource name can be defined as a transaction in one system and a logical terminal (LTERM) in another.
- IMS does not prevent a logical terminal with the same name to be active in two or more IMS systems concurrently.

It is an IMS system programmer responsibility to ensure that resource definitions and usage are consistent within a shared queues group. Some aids to consistency are:

- Cloning: If system definitions are cloned among all members of a shared queues group, for example:
  - A name will not be used as a transaction code in one system and a logical terminal name in another.
  - A non-conversational transaction defined to one system will not be defined as a conversational transaction in another.
- Static Definitions: If resource definitions (transactions and logical terminals) are statically defined as well as cloned, the probability of executing with consistent resource definitions is greatly enhanced.

Resource names can be dynamically created. The use of this capability offers the opportunity for error. With shared queues, transactions can be dynamically created with the assistance of the Output Creation Exit. We recommend that all transactions be statically defined. Changes in transaction definitions (add, delete, alter) can be introduced through the online change process.

Extended Terminal Option (ETO) users can dynamically create logical terminals. ETO with shared queues is discussed in the section 8.4, "ETO Considerations with Shared Queues" on page 59.

## 10.5 MSC Considerations

IMS members of a shared queues group can participate in routing and processing message traffic related to an MSC network. Changes made to MSC in support of IMS/ESA V6 and shared queues, in particular, must be understood and reflected in any implementation plan.

MSC network definition limits have been changed with IMS/ESA V6 and are as follows:

- MSPLINK macro statements from 255 to 676. Increases the number of physical links that can be defined to an IMS.
- MSLINK macro statements from 254 to 676. Increases the number of logical links that can be defined to an IMS.
- MSNAME macro statements from 255 to 676. Increases the number of logical link paths that can be defined to an IMS.
- SYSID specifications from 255 to 2036. Increases the number of unique SYSIDs that can be defined among the IMS systems in an MSC network.

These new limits allow network definitions to be expanded. If pre-V6 systems are to be members of an MSC network that includes IMS V6 systems, the MSC network definitions should adhere to the pre-V6 limits.

### 10.5.1 Inter-IMSplex MSC Without Shared Queues

Inter-IMSplex MSC links are connections between IMS systems within the IMSplex and IMS systems outside the IMSplex. For example, one might have had an IMS system in Dallas connected to an IMS system in Chicago through MSC, and the Dallas IMS system will be converted to a cloned IMSplex.

The cloning of the Dallas IMS system may or may not have an impact on the system definition for the Chicago IMS system. Let us assume that the original IMS system in Dallas had an MSC SYSID of 10 and the Chicago IMS system had an MSC SYSID of 20. The original Dallas system definition (SYSID 10) contained information about the Chicago SYSID of 20, and the Chicago system definition (SYSID 20) contained information about the Dallas SYSID of 10.

We are now going to redefine the Dallas system into two IMS systems, one with an MSC SYSID of 10 and the other with an MSC SYSID of 11.

IMS MSC rules require that all client IMS systems (originators of IMS transactions) need to be defined to the server IMS system (processor of IMS transactions) and that the server IMS system needs to be defined to its client IMS systems.

Thus, if end-users connected to both Dallas systems need to send transactions, through MSC, to be processed in the Chicago system, both Dallas systems (the clients) need to be MSC capable and have information about the Chicago SYSID of 20 in their system definitions. In addition, the Chicago system must have information about both Dallas SYSIDs (10 and 11) in its system definition. Thus, the addition of another MSC definition to Dallas system forces a system definition change to the Chicago system. This is true even if there is just one physical MSC link connecting the Chicago system to the Dallas IMSplex.

On the other hand, if all MSC transactions originated from the Chicago system (the client) and were processed in Dallas, one could get by with just defining one of the Dallas systems (the server) to the Chicago system. Of course, this would imply that all the transactions originated in Chicago could process only on one of the Dallas systems. While this type of configuration would minimize system definition changes in Chicago, it would negate the availability benefits of having multiple cloned IMS systems (servers) in Dallas.

### 10.5.2 Migration to Shared Queues

When an MSC network includes one or more members of a shared queues group, the following changes or new function must be understood:

- MSC physical and logical links can be defined among the IMS systems in a shared queues group. These links can only be used (started) when one or more of these systems is/are outside the shared queues group. For example, IMSA and IMSB are members of the same shared queues group and have a physical and logical MSC link defined that would allow messages to be sent between the two systems if the link were started. This link can be started only when IMSA or IMSB (or both) is (are) removed from the shared queues group.

  If the MSC link cannot be started between IMSA and IMSB, when both are in a shared queues group, why would one ever define it? One reason for the link definition is to allow an orderly migration of IMSA and IMSB to the shared queues environment.

- If any member of a shared queues group has a real MSC link to a system remote to the shared queues group, all members of the shared queues group must have MSC defined. When a system has MSC defined, the system definition process includes MSC-related code to support MSC message traffic. This MSC support is required in all members of the shared queues group that process messages received from or sent to an MSC destination.

  Since it is likely that any member of the shared queues group will process a message received from an MSC system remote to the group, all members of the group are required to have MSC generated to allow each of them to properly process a message which includes an MSC extension to the message prefix. If MSC is not generated for a system and that system receives a message containing an MSC extension to the message prefix, the message will not be properly processed, and the result of the error is unpredictable.

  The safe course of action is to include MSC definitions in the system generation process for each IMS member of the shared queues group to guarantee messages are always processed properly. The inclusion of MSC capability requires one of each of the following macros in the system generation input: MSPLINK, MSLINK, and MSNAME. If all systems in the shared queues group are clones of one another, a single system definition and generation (including MSC definitions) can be used to create a single RESLIB that is shared among all members of the group and all are capable of processing MSC-related message traffic.

- The rules for the specification of SYSIDs have been slightly changed. Listed below are the existing rules along with the changes:
  - A SYSID must be unique to a given IMS system in the MSC network and must be known as that system′s ′local SYSID′. This is an existing rule.
  - An IMS may have more than one local SYSID defined. This is actually required if parallel sessions are to be used (A unique local SYSID for each parallel session is required). This is an existing rule.
  - Local and remote SYSIDs defined to members of a shared queues group are shared among the members of the group. This ′sharing′ of SYSIDs defined among the members of the group enables the routing of messages among the members of the group to and from the remote systems that have MSC connections to one or more members of the group.

    For example, IMSA has a local SYSID of 10 and MSC definitions that allow it to establish an active logical link with IMSR (local SYSID of 11) which is remote to the shared queues group. IMSB is also a member of the shared queues group with a local SYSID of 20 which allows it to establish an active logical link with IMSS (local SYSID of 21), which is also remote to the shared queues group. For simplicity′s sake, there are no other members of the shared queues group. We also assume that IMSA and IMSB have not cloned their definitions.

```
     IMSA                           IMSB

  AR    MSPLINK  NAME=IMSR        BS    MSPLINK  NAME=IMSS
        MSLINK   PARTNER=AA             MSLINK   PARTNER=BB
  ATOR  MSNAME   SYSID=(11,10)    BTOS  MSNAME   SYSID=(21,20)
```

    When IMSA and IMSB are both active as members of the same shared queues group, they exchange SYSID tables to inform each of the other′s local and remote SYSIDs. IMSA knows that SYSID 21 is a valid destination SYSID that is remote to the shared queues group, and SYSID 20 is a valid SYSID local to the shared queues group. Similarly, IMSB knows that SYSID 11 is a valid destination SYSID that is remote to the share queues group, and SYSID 10 is a valid SYSID local to the shared queues group.

    In addition, IMSA and IMSB, because their system definitions are not cloned, dynamically create MSNAME definitions (build the control block required to support an MSNAME definition (LNB)) for MSNAMEs defined to other members of the shared queues group. The definitions for IMSA and IMSB are as follows after both have joined the group:

```
    IMSA                            IMSB

 AR    MSPLINK  NAME=IMSR        BS    MSPLINK  NAME=IMSS
       MSLINK   PARTNER=AA             MSLINK   PARTNER=BB
 ATOR  MSNAME   SYSID=(11,10)    BTOS  MSNAME   SYSID=(21,20)

 BTOS  MSNAME   SYSID=(21,10)    ATOR  MSNAME   SYSID=(11,20)

       Note:  The local SYSID for a dynamic MSNAME is set to
              the lowest local SYSID actually defined in the
              IMS system where the MSNAME is created.
```

The dynamically created MSNAMEs allow messages originated on IMSA with a directed routing destination of BTOS to be successfully queued to the remote ready queue. If IMSB has an active logical link with IMSS, it retrieves messages placed on the remote ready queue by IMSA and sends them to IMSS. A rewording of the previous two sentences would state the reverse with regard to messages originating on IMSB with a directed routing destination name of ATOR.

Remote transaction definitions cannot be used with dynamic MSNAMEs because the system definition process will fail if a remote SYSID specified on a TRANSACT macro cannot be matched to the same remote SYSID in an MSNAME macro. As an alternative to statically defining remote transactions and 'matching' logical link paths (MSNAME macros), the Output Creation Exit is enhanced in IMS/ESA V6 to dynamically create (as an exit option) remote transactions. This allows transaction input not defined to one system to be queued to the shared queues. When the logical link to the remote IMS is active, the IMS in the shared queues group with the active logical link retrieves these remote transactions from the shared queues and sends them to the remote IMS for processing.

On the other hand, remote LTERMs can not be created by the Output Creation Exit to allow messages destined for a terminal remote to the shared queues group to be queued to the shared queues. Also, keep in mind that online change does not support remote LTERMs. When remote LTERMs are defined to a member of the shared queues group as the means to route messages to an IMS system remote to the group, the other members of the group must either (a) include these remote LTERM definitions in their system definition input (this is automatic if the systems are cloned) or MSC descriptors or (b) provide the appropriate exit routines to route remote LTERM messages by specifying the remote SYSID or MSNAME as the destination.

When remote transactions and/or logical terminal definitions are used to route messages from a member of a shared queues group (IMSA, for example) to an IMS (IMSR, for example) remote to the shared queues group, the other members of the shared queues group:

- Must have cloned IMSAs remote transaction and logical terminal definitions in their system definitions
- May use the Terminal Routing or Program Routing exits to set the message destination name to the MSNAME or SYSID defined in the IMS (IMSA) that is associated with the real logical link to IMSR,

  - For remote transactions, may use the Output Creation Exit to set the destination to be a remote transaction defined in IMSA

- The use of directed routing among members of a shared queues group is not supported. That is, directed routing can not be used to route a transaction or message from one IMS in the shared queues group and have it processed as a local transaction or message by another member of the group. This restriction arises because transactions and messages sent with directed routing have an MSNAME as the destination name and are queued to the Remote Ready Queue on the shared queues. Since an MSC link can never be started between two members of a shared queues group, the partner IMS in the shared queues group never registers interest in any MSNAME local to the shared queues group.

  In general, the preceding description of the migration problem, when directed routing is used among the members of a shared queues group, might be solved in a couple of different ways depending upon where the directed routing destination name is set:

  - If set by the application program, the application programs that use directed routing can be changed to not set the MSNAME destination. Instead, the programs simply set the destination name to the transaction code or logical terminal name for these messages. These messages are then queued to the appropriate Ready Queue (Transaction or LTERM) and will be processed by any IMS in the shared queues group with interest in the destination name. This alternative might be undesirable because it requires that application programs be changed.

  - A change introduced with the fix to APAR PQ13065 gives the Program Routing Exit (DFSCMPR0) the ability to change the destination name set by an application program′s CHNG call. Previously, the other MSC routing exits (Input Message Routing (DFSNPRT0) and Link Receive (DFSCMLR0) Exits were able to change a destination name.

    With IMS V6 and this APAR fix applied, the Program Routing Exit can override a directed routing request on a CHNG call to a modifiable alternate TPPCB. That is, when an application program issues a CHNG call with the I/O area set to an MSNAME, the Program Routing Exit is called and can request IMS to override the MSNAME destination with the destination name in the I/O area specified when an ISRT message segment call is issued. IMS will then insert the message to the LTERM/TRANCODE name in the message segment instead of the MSNAME that was specified by the application program in the CHNG call.

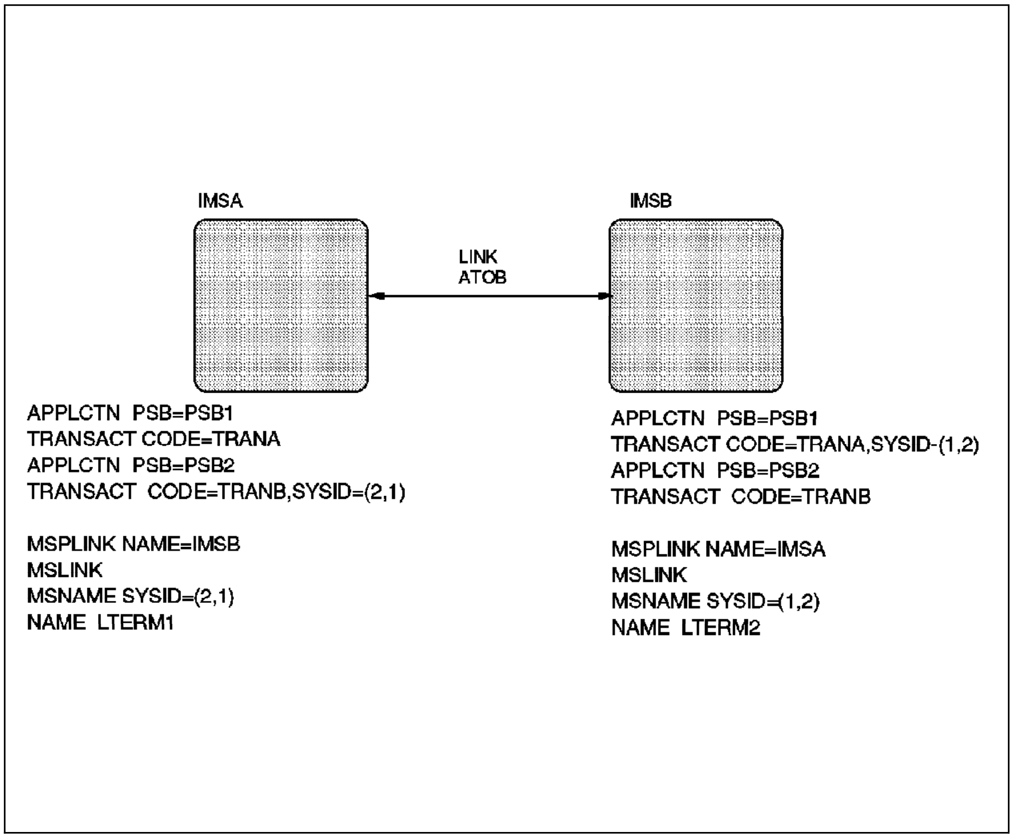
    The LTERM/TRANCODE must be defined in the IMS where the application program is executing, or use the Output Creation Exit to create a dynamic transaction or LTERM.

    The use of the fix to APAR PQ13065 eliminates the requirement to change application programs.

    Application programs that use non-modifiable, alternate TPPCBs with an MSNAME destination need to be changed to specify the LTERM/TRANCODE name that is now local to the shared queues group.

## 10.5.3 Example: Migration of Two MSC-Connected Systems

Both IMSA and IMSB already exist and are connected through an MSC link as shown in Figure 11.



```
APPLCTN  PSB=PSB1
TRANSACT CODE=TRANA
APPLCTN  PSB=PSB2
TRANSACT  CODE=TRANB,SYSID=(2,1)

MSPLINK NAME=IMSB
MSLINK
MSNAME SYSID=(2,1)
NAME  LTERM1
```

```
APPLCTN  PSB=PSB1
TRANSACT CODE=TRANA,SYSID-(1,2)
APPLCTN  PSB=PSB2
TRANSACT  CODE=TRANB

MSPLINK NAME=IMSA
MSLINK
MSNAME SYSID=(1,2)
NAME  LTERM2
```

*Figure 11. Two IMS systems With an MSC connection*

The overall plan is to move both IMSA and IMSB to a unique shared queues group. A practical shared queues implementation plan may be selected that first moves IMSB into the shared queues group as shown in Figure 12 on page 96. The move of IMSB to a shared queues group does not require any system definition changes for IMSA or IMSB.

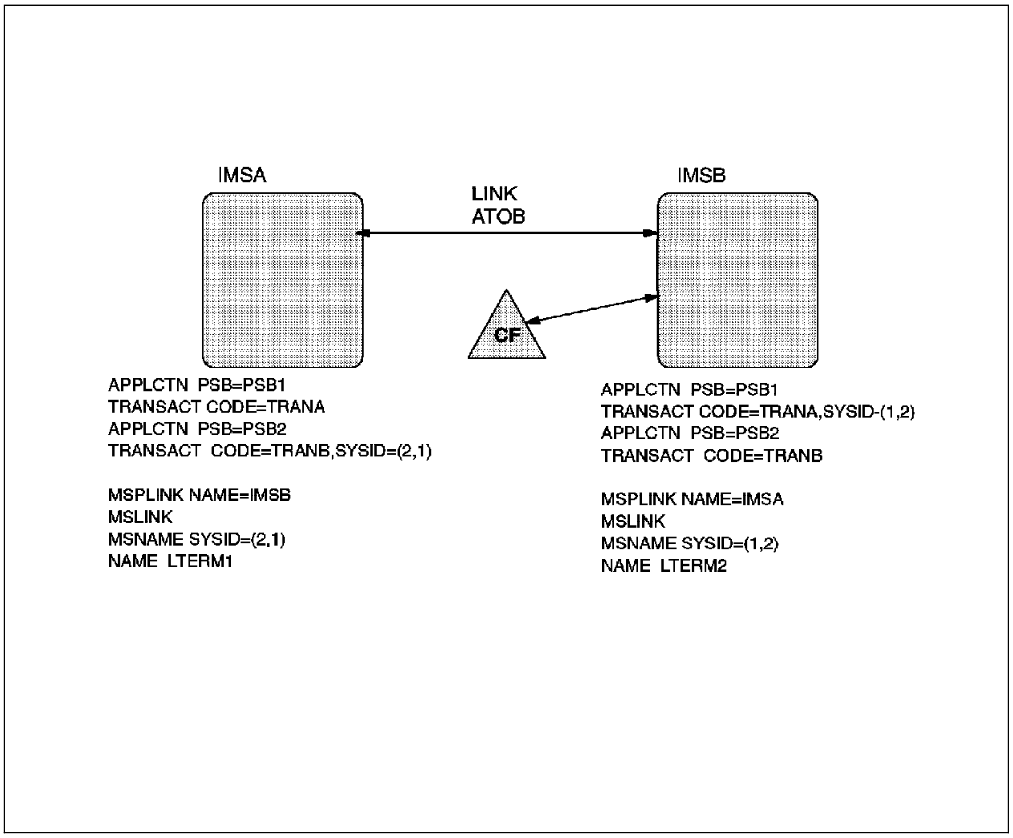


*Figure 12. Migration of IMSB to a Shared Queues Group*

Once the move of IMSB is made, the plan requires that IMSB′s execution within the shared queues group be established over some period of time before IMSA is moved to the same shared queues group. Prior to IMSA joining the shared queues group, the MSC link between IMSA and IMSB is required. Once IMSA joins the shared queues group, the MSC link can no longer be started and, in fact, is not needed. The shared queues provide the message routing function. Shared queues are a replacement for MSC links once the MSC partners join the same shared queues group. In Figure 13, IMSA and IMSB have been migrated to a Parallel Sysplex environment where both are members of the same data sharing and shared queues groups.

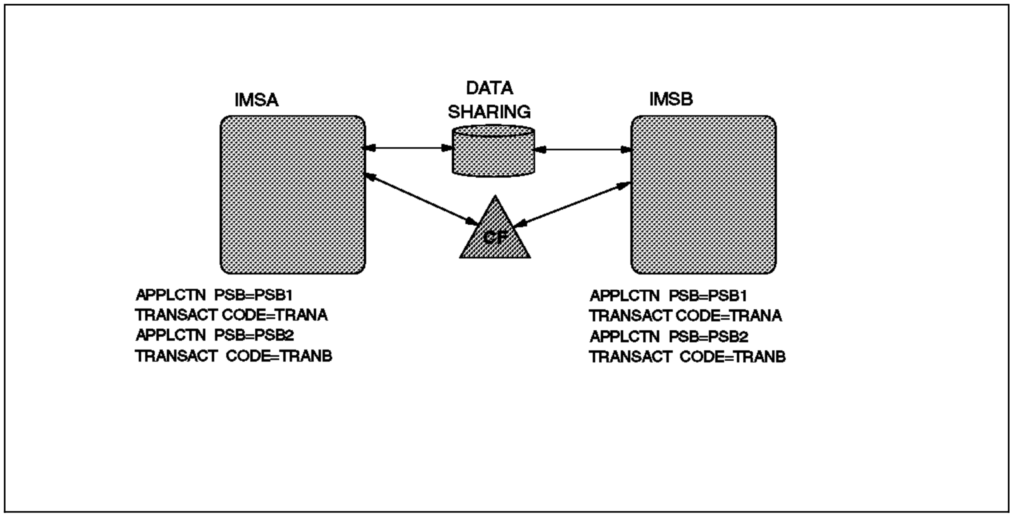


*Figure 13. Migration of IMSA and IMSB to a shared queues group is completed*

Note the system definition changes that were made when IMSA joined the data-sharing and shared queues groups with IMSB (Refer to Figure 13):

- The remote transaction definitions have been changed to local definitions. The systems are now cloned from a transaction processing perspective.
- The MSC link definitions have been removed because they are no longer required. The removal of MSC definitions is not supported by a MODBLKS system definition with a subsequent online change. Therefore, both systems must be bounced to incorporate the removal of these definitions.
- The remote logical terminal (LTERM1 in IMSA and LTERM2 in IMSB) definitions have been removed. Given that terminal access to the shared queues group is to be allowed through any member of the group:
  - If dynamic terminals (ETO-created) are used with IMSA and IMSB, both systems must be set up to create the terminal-related control blocks to support sessions using LTERM1 and LTERM2.
  - If static terminal definitions are used, the terminal-related system definition macros of both systems must be merged into each systems, Stage1 input. Of course, if the systems are to be cloned, they can be merged into a single Stage1 to produce a single RESLIB to be shared between IMSA and IMSB. The changes required to merge the two systems terminal-related macros requires that both systems must be bounced to incorporate the static terminal definition changes.

If directed routing as well as remote transaction and logical terminal definitions is used to route transactions and messages between IMSA and IMSB, the final migration step may involve additional considerations.

If applications are setting the directed routing destination name either:

- They must be changed to set the destination name for their CHNG or ISRT calls to the local (previously defined as remote) transaction or logical terminal name.
- A Program Routing Exit (DFSCMPR0) must be implemented to change the application provided MSNAME to the now locally defined transaction or logical terminal name. This use of the Program Routing Exit requires the fix to APAR PQ13065.

If the Program Routing Exit sets the remote destination name, it must be changed to not set this name and allow the application destination name specification to stand. This assumes that the Program Routing Exit will be driven. Based on the example of migrating IMSA and IMSB to a shared queues environment and eliminating the MSC definitions in both systems, the exit is no longer called for application CHNG and ISRT calls. When this is true, the migration plan for IMSA and IMSB simply removes the Program Routing Exit from both systems.

### 10.5.4 Example: MSC Link from a Shared Queues Group Member to a Remote IMS

In Figure 14 on page 98, an MSC link is shown from IMSA in the shared queues group to IMSC, which is remote to the shared queues group.

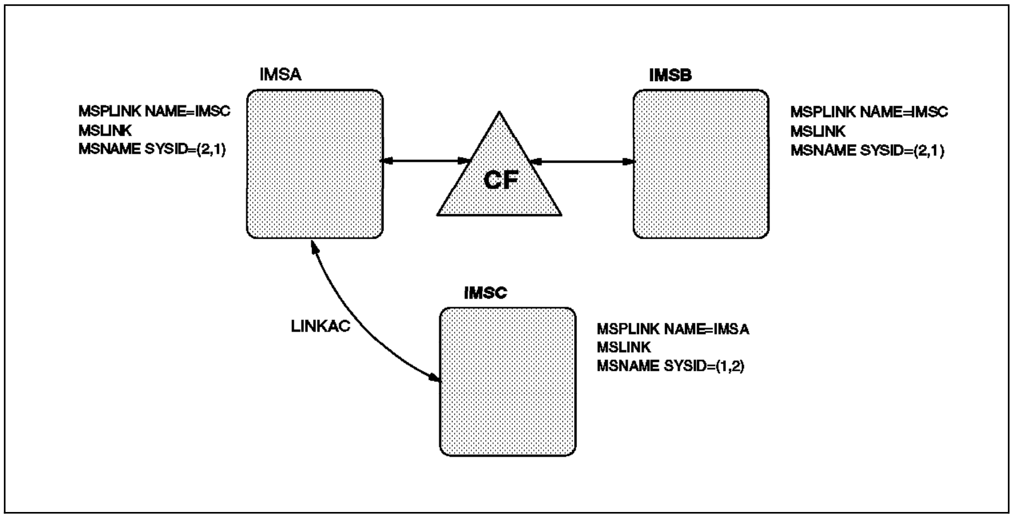


*Figure 14. MSC link from shared queues group member to remote IMS*

A few things associated with this figure should be noted.

- The MSC definitions for IMSA and IMSB are identical. Given that IMSA and IMSB are clones of one another, a single system definition can be used to generate both systems. Another way of saying the same thing is that IMSA and IMSB can share the same RESLIB.
- The single MSPLINK macro definition in IMSC points to the APPLID of IMSA. This limits the configuration shown to a single physical/logical link between IMSA and IMSC.
- Remote transaction and logical terminal definitions in IMSA and IMSB can also be cloned. This allows, for example, any terminal in session with either IMSA or IMSB to enter a transaction that is defined as remote, have it queued in the coupling facility (Transaction Ready Queue), and delivered to IMSC by IMSA for processing whenever the logical link between IMSA and IMSC is active.
- Similarly, transactions defined as remote to IMSC can be defined as local to both IMSA and IMSB. Whenever IMSC receives a transaction defined to it as remote, IMSC sends the transaction to IMSA. IMSA queues the transaction in the Coupling Facility, and it is processed by either IMSA or IMSB.
- Reply messages to transactions received from IMSC which have been generated by either IMSA or IMSB are queued in the Coupling Facility to the MSNAME associated with the physical/logical link to IMSC, delivered to IMSC by IMSA, and sent to the terminal attached to IMSC that entered the original remote transaction.

  Similarly, reply messages generated within IMSC by remote transaction input received from IMSA over the MSC link are delivered back to IMSA through the MSC link, queued in the coupling facility to the LTERM Ready Queue, and delivered to the terminal associated with the logical terminal destination name whether in session with IMSA or IMSB.

## 10.5.5 Example: MSC Link Between Members of Two Shared Queues Groups

In Figure 15, an MSC link is shown from IMSA in the shared queues group to IMSD in another shared queues group. The MSC definitions for IMSA and IMSB are cloned as are the MSC definitions for IMSC and IMSD in the other shared queues group.

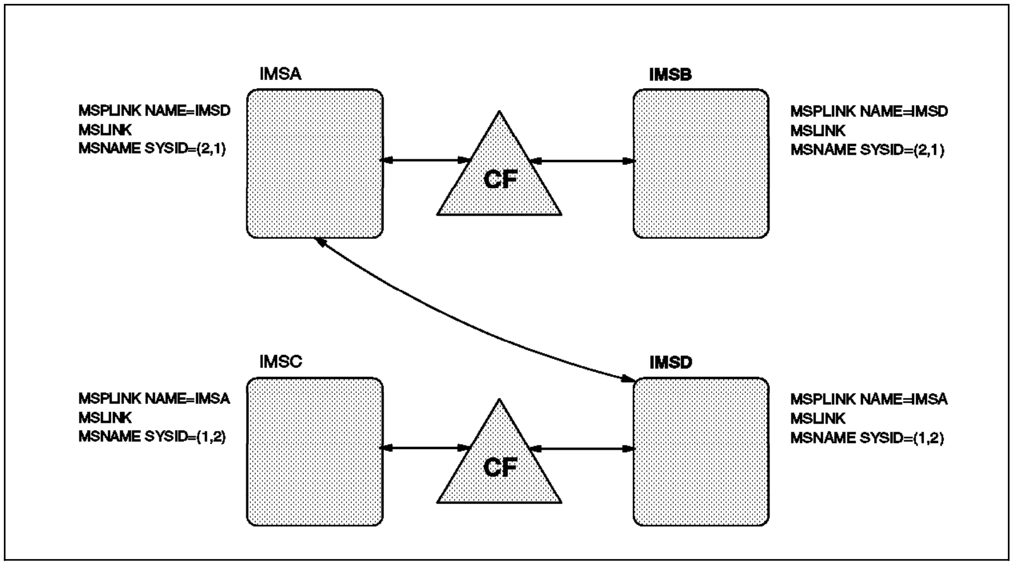


*Figure 15. MSC links between members of two shared queues groups*

Comments follow:

- Terminals attached to any of the four IMS systems can enter remote transactions that are sent to the other shared queues group through the defined MSC physical/logical link.
- A transaction received over the MSC link can be processed by any IMS in the shared queues group that has a local definition for that transaction.
- The local SYSIDs of the members in each Parallel Sysplex configuration are the same. The only allowable MSC connection, nevertheless, is between IMSA and IMSD.

## 10.5.6 Example: Multiple MSC Links between Members of Two Shared Queues Groups

In Figure 16 on page 100, two remote links are shown, one from IMSA to IMSC and the other from IMSB to IMSD.

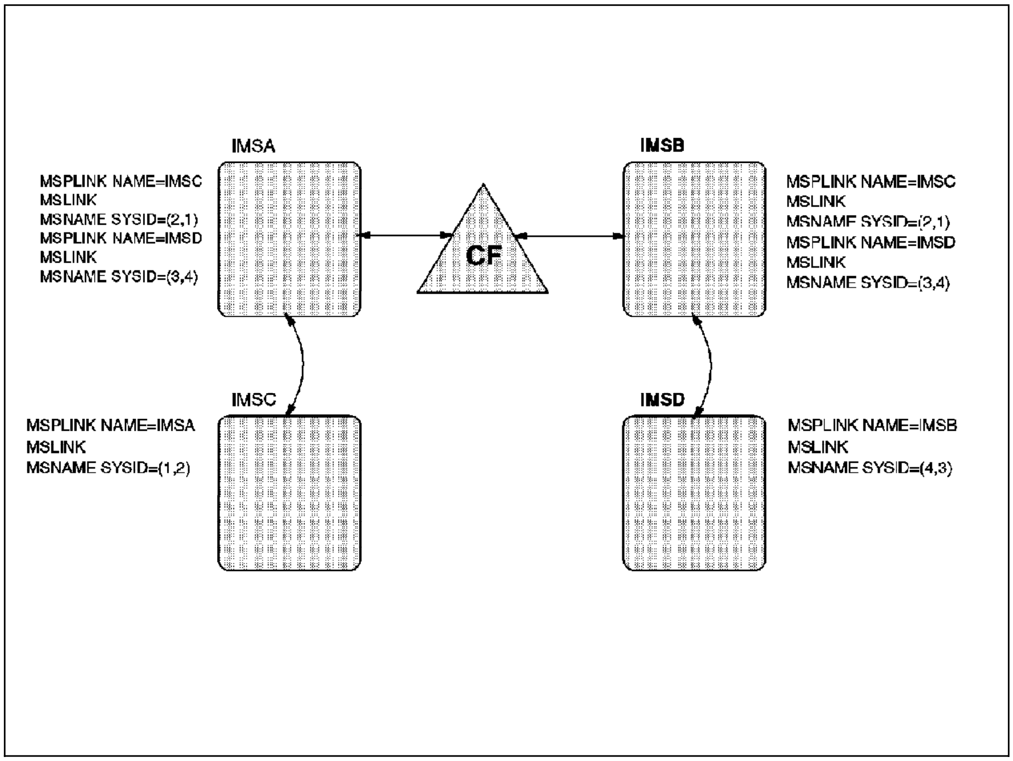


*Figure 16. MSC connections from shared queues group members to remote IMS systems*

Comments follow:

- The MSC definitions for IMSA and IMSB have been cloned.
- Only two links can be established and are as shown because they are the two links that have MSPLINK definitions that correspond (point to) among the IMSs shown.

## 10.5.7 Example: Backup MSC Link Configuration

In Figure 17 on page 101, the MSC definitions are set up so that a backup link between IMSC and IMSB can be started if the link between IMSA and IMSC should fail for whatever reason. This is a reasonable configuration in the event the link between IMSA and IMSC fails and cannot be restarted immediately (in a timely fashion).

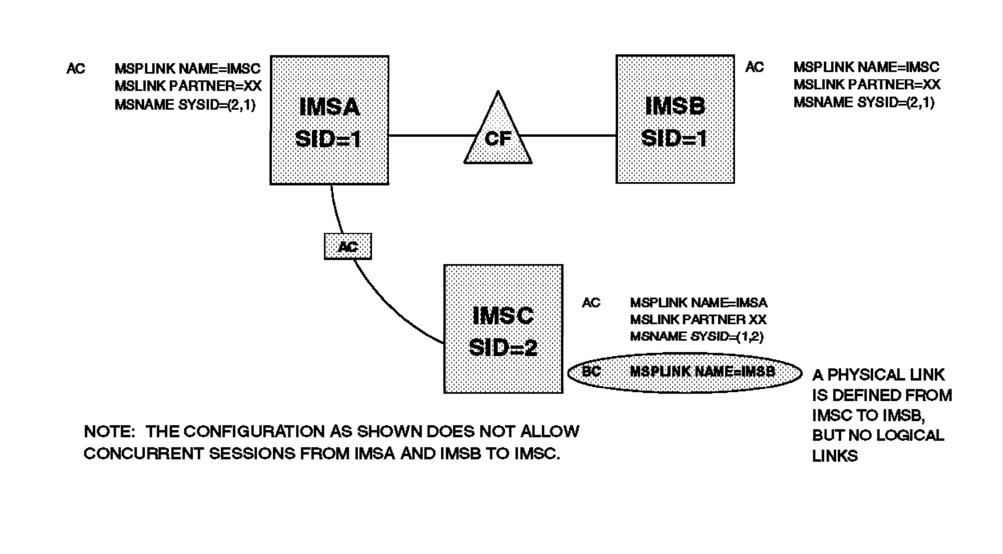


*Figure 17. Backing up logical links from a shared queues group to a remote IMS*

Comments follow:

- Note the MSC definitions for IMSC include two MSPLINK macros, one that points to IMSA and the other to IMSB.
- The definitions as shown do not allow concurrent active links from IMSA and IMSB to IMSC.
- The MSC definitions for IMSA and IMSB are clones of one another.

In Figure 18, the MSC link has failed between IMSA and IMSC.

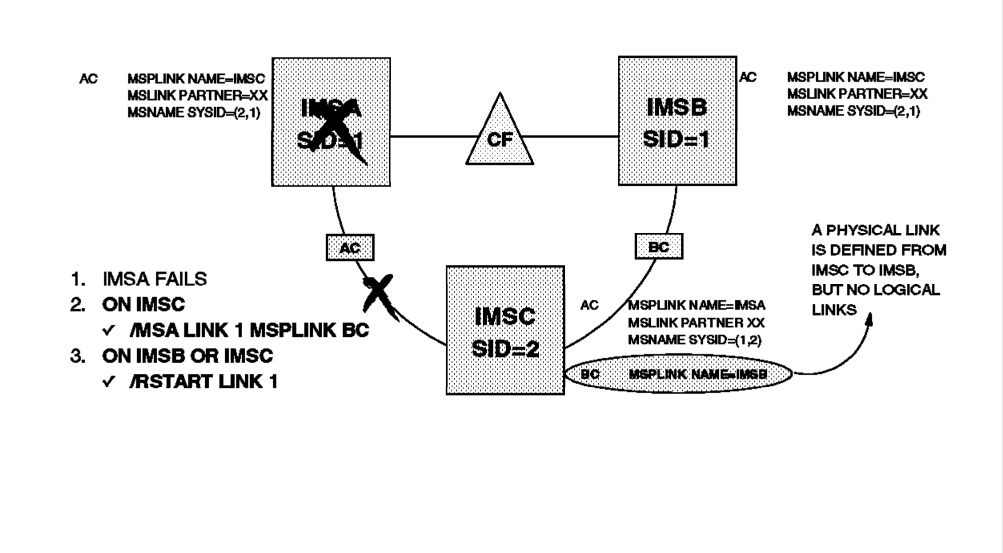


*Figure 18. Starting a backup link to a shared queues group*

Starting a link between IMSC and IMSB requires a couple of commands to be issued:

```
                  On IMSC:  /MSA LINK 1 MSPLINK BC
        On either IMSC or IMSB:  /RSTART LINK 1
```

Depending upon your MSC definitions and the state of the failed link, one can have to issue other commands before the backup link can be started,for example with the /PSTOP LINK and /CHANGE link FORCESS COLDSESS commands.

## 10.6 Serial Transactions

Some transactions must reside on the same IMS subsystem to process transactions as SERIAL to guarantee that they are processed on the local IMS subsystem that has that transaction defined.

### 10.6.1 Serial Transactions With Traditional Queuing

One of the options on the TRANSACT macro is SERIAL=YES or NO. The default is NO. SERIAL=YES causes IMS to perform serial processing of messages for a given transaction type. In other words, transactions of a given type are guaranteed to be processed in the exact order of arrival (enqueue).

SERIAL=YES only effects scheduling and operation within a single IMS system. As such, one must perform some analysis if one currently has SERIAL=YES transactions.

If the intent of SERIAL=YES is to force serial processing of transactions entered from the same end user, but there is no requirement for transactions from multiple end-users to be processed in the order of arrival, one can clone the transaction as SERIAL=YES on all the IMS systems within the IMSplex as long as one ensures that the SERIAL=YES transactions entered by an end-user connected to a particular IMS system remained within and were processed within that particular IMS system. Shared queues enforces this type of processing for SERIAL=YES transactions.

If the intent of SERIAL=YES is to force serial processing of transactions entered from multiple end-users, one has to simulate SERIAL=YES across the entire IMSplex. This can be done by defining the transaction on one of the systems as a local transaction with SERIAL=YES specified and defining the transaction on all of the other systems as a remote transaction. It should be noted that this technique only approximates IMSplex-wide SERIAL=YES processing. If two end-users connected to two different IMS system within the IMSplex enter the same transaction type at approximately the same time, there is no guarantee that the two transactions will be enqueued on the “local” system (the system that has the transaction defined as local) in the same order as they arrived within the IMSplex. This is because one or both of the transactions had to flow across an intra-IMSplex MSC link to the local IMS system.

### 10.6.2 Serial Transactions With Shared Queues

Without shared queues, transactions defined as SERIAL=YES on the TRANSACT macro are processed in the sequence of arrival by the IMS which received the input transaction. If a serial transaction accesses unavailable data base resources, it is:

- Abended with a U3303 abend code.

- Requeued to the head of the queue for the transaction.
- The transaction is USTOPPED.

Later on, when the transaction is started, the original processing sequence is preserved.

The preceding actions occur unless the application program has issued an INIT STATUS GROUPA call and can successfully handle BA or BB status codes.

With shared queues and serial transactions:

- A serial transaction is always processed by the front-end IMS— that is, by the IMS that received the input transaction. Serial processing of transactions is guaranteed only within the front-end IMS.
- The same transaction code, defined as SERIAL=YES to multiple IMS subsystems within the same shared queues group, can not execute serially among the members of the group.

Prior to implementing shared queues, one must identify if any transactions are defined as serial. If some are, determine whether the SERIAL=YES specification is required. This determination might not be possible because ′no one knows′ why a given transaction was defined as SERIAL=YES. If it is determined that SERIAL=YES is not necessary for a given transaction, the transaction definition can be changed to SERIAL=NO.

If serial transaction processing is important among the members of a shared queues group, the following setup and operational procedures can be used to provide serial processing for a transaction in the order multiple transaction instances are queued to the coupling facility:

- Define the transaction as non-serial (SERIAL=NO on the TRANSACT macro).
- Define the APPLCTN macro associated with the transaction as SCHDTYP=SERIAL. This prevents the application program from being scheduled into more than one dependent region within a given IMS simultaneously.
- Assign the transaction to a CLASS that only executes on one IMS.
- Use the Non-Discardable Message Exit (DFSNDMX0) to requeue and USTOP any transaction that abends with a U3303.

## 10.7 Undefined Destinations

When a message is to be queued, IMS code requires that an internal control block exist with the same destination name as that of the message. In general, the control block represents the destination of the message, which is either a transaction or a logical terminal. In a shared queues environment the destination can be not be defined to the system that is required to enqueue the message. If the destination is not defined on that system, the real control blocks representing the destination are not available. If the real control blocks are not available, one or more control blocks are created and used to enqueue the message.

## 10.7.1 Destination Determination

Before placing a message on a queue, IMS tries to locate the control block representing the destination of the message. If a local control block is not found, the destination is unknown to that IMS. If the Extended Terminal Option (ETO) is active (ETO=Y) or shared queues is active (SHAREDQ=xxx), the Output Creation Exit (DFSINSX0) is called to allow the user to identify the unknown resource. The exit must indicate whether the resource is a transaction, LTERM (requires ETO to be active), remote transaction, or whether the resource is invalid. IMS creates real control blocks for those destinations the exit identifies as valid.

## 10.7.2 Back-End Processing of Input Transactions

For back-end processing, when an application program issues a GU to the IOPCB, IMS tries to locate the control block representing the input LTERM. If the input LTERM is defined to the back-end system, the real control blocks are used to enqueue the response message on the shared queues. If the input LTERM is not defined to the back-end system, a control block is created and is used by the application program executing in the dependent region to enqueue reply messages on the shared queues.

If an application inserts a message to an alternate TPPCB and the destination is not defined in the system, but the Output Creation Exit indicates it is a valid destination, IMS creates the required control block(s) to represent the destination. If the exit identifies the resource as an LTERM (ETO=Y required), a real user structure is created and used to enqueue the message. If the back-end system is not ETO-capable and the Output Creation Exit does not exist, by default IMS does not create control blocks for alternate TPPCB activity (CHNG or ISRT calls). The call is rejected. If the back-end system Output Creation Exit designates the destination name to be a transaction, a control block (SMB) is created and is used for purposes of queuing messages to the shared queues.

## 10.7.3 Comments and Recommendations

For ETO users, the occurrence of undefined destinations by a back-end system to support IOPCB activity cannot be avoided and does not require the use of the Output Creation Exit to support transaction input and reply output.

If static terminal definitions are used and they are cloned among all IMS systems in the shared queues group, there is no need for a back-end IMS system to create LTERM-related control blocks to support IOPCB activity. When static definitions are not cloned and the input LTERM is not known to a back-end system, control blocks (CNTs) are created and used to support the application program′s IOPCB activity.

For alternate PCB activity (CHNG and ISRT calls) by an application in a back-end system:

- ETO users can use the Output Creation Exit to create user structures whenever the destination name is considered to be an LTERM. To avoid confusion over whether a destination name is an LTERM versus a transaction, it is recommended that ETO users clone their transaction definitions among all members of a shared queues group. This ensures that the Output Creation Exit is only driven for LTERM output.
- Static terminal users are advised to clone their transaction and terminal definitions in a shared queues environment. If this is done, there will never

be a need to dynamically create control blocks to support alternate TPPCB program to program switch activity.

- For all IMS members of a shared queues group, whether using dynamic terminals (ETO=Y) or static terminal definitions, the addition, deletion, or change of transaction definitions should be accomplished through the use of online change.

## 10.8 Online Change

Online change can be used with shared queues to make changes to IMS systems while they are executing. The use of online change in a Parallel Sysplex is explained in 18.8, “Online Change Procedures” on page 184.

## 10.9 SPOOL Terminals

The term “SPOOL” refers to IMS terminals that are defined in the system definition as DISK, PUNCH, PRINTER, TAPE, and/or SPOOL. In other words, we are talking about terminals that are really MVS sequential output files. Since these terminals (output files) cannot be shared, one must take SPOOL terminals into account when cloning an IMS system.

If the only use for SPOOL terminals within the existing system is for secondary master output, with no application output directed to the terminal, one would probably use the same SPOOL terminal definition for each system. However, one should specify unique LTERM names for each system (see 10.1, “Master Terminals With IMS/ESA Version 5” on page 87 or 10.2, “Master Terminals With IMS/ESA Version 6” on page 88). This should not cause any application problems since application programs are not sending output to the SPOOL terminal.

If the SPOOL terminal is used for application output (for example, an audit trail or journal), the LTERM name is known to these applications and is not easily changed. With shared queues, multiple systems cannot have different terminals with the same LTERM name. This means that a consolidated SPOOL terminal will be used for this application. If shared queues are not used, multiple IMS systems can use the same LTERM name. The application could use the same LTERM name to write to multiple SPOOL terminals, one on each IMS system.

If multiple SPOOL terminals are used, one would have multiple audit trails or journals. This might have an application impact on the non-IMS programs that process the SPOOL output. One could write a program that merges the multiple SPOOL data sets into a single, consolidated data set, but the records in the individual SPOOL data sets would have to contain some control field (date/time for example) in order to merge the individual data sets. If such a field did not exist, the IMS physical terminal output edit routine, DFSCTTO0, could be used to add a date/time field for merging.

If a single consolidated SPOOL terminal is used for application output, there will be minimal impact to the programs that process the SPOOL data set. This is easily handled with shared queues since the SPOOL terminal would be defined on one system, and all messages for the terminal would be processed by this system. If shared queues are not used, intra-IMSplex MSC facilities could be used. One would define the SPOOL terminal as a local terminal on one of the IMS systems and as a remote terminal on all the other IMS systems.

We recommend the use of a single, consolidated set of SPOOL terminals since using the same LTERM name concurrently in multiple IMS systems within the IMSplex is not allowed with shared message queues.

For information on the IMSWTnnn procedure used with Spooled SYSOUT terminals and its impact on system definitions, see 17.2, “IMS Jobs” on page 170.

# Part 4. Data-Sharing and Shared Queues Considerations

# Chapter 11. Data-Sharing Enablement

If data-sharing is not currently enabled in the existing environment, there are a number of tasks that must be accomplished to enable it in the target environment.

Information on understanding IMS data-sharing is available in the IBM *IMS/ESA Block Level Data Sharing* class and in publication, *IMS/ESA Data Sharing in a Parallel Sysplex*, SG24-4303.

## 11.1 IMS System Definition

The following parts of your IMS system definitions should be reviewed.

- IMSCTRL macro
  - IMSCTRL ... IMSID=

    The specification of IMSID on the IMSCTRL macro sets the default IMS subsystem ID for this definition. It might be overridden at execution time by the IMSID parameter. An installation might want to define a generic IMS name, such as *IMS* in the system definition and override it with a specific name at execution time.
  - IMSCTRL ... DBRCNM=

    The specification of DBRCNM on the IMSCTRL macro sets the default name for the DBRC address space procedure. It can be overridden at execution time.
  - IMSCTRL ... DLINM=
  - IMSCTRL macro

    The specification of DLINM on the IMSCTRL macro sets the default name for the DL/I address space procedure. It can be overridden at execution time.
  - IMSCTRL ... IRLM=Y

    The specification of IRLM=Y on the IMSCTRL macro sets the default for batch jobs. This can be overridden in batch JCL.
  - IMSCTRL ... IRLMNM=

    The specification of IRLMNM on the IMSCTRL macro sets the default name for the IRLM subsystem to which this IMS will connect. It can be overridden at execution time.

  Consider specifying generic names for the IMSID, DBRCNM, DLINM, and IRLMNM on the IMSCTRL macro. The DFSPBxxx member for each IMS can specify the actual names to use. This allows one to easily clone systems using one definition.
- DATABASE ... ACCESS=UP

  The default for the ACCESS parameter is EX (exclusive). If a database is to be shared, this must be changed. ACCESS=UP allows the system to update the database.

  The access intent can be changed at execution by using the `/START DATABASE ... ACCESS=xx` command.

- APPLCTN ... SCHDTYP=SERIAL

  SCHDTYP=SERIAL, which causes a PSB to be scheduled in only one dependent region or DBCTL thread at a time, is enforced only within a single IMS. If the PSB is defined to multiple IMSs, it can be scheduled in multiple dependent regions or threads at the same time. The reasons for this specification must be reviewed to determine whether that APPLCTN can be defined in all IMSs or whether some application changes might have to be made.

- TRANSACT ... SERIAL=YES

  This specification forces serial processing (FIFO) of messages within a single IMS TM system. As with the SCHDTYP parameter on the APPLCTN macro, it is enforced only within a single IMS, and the reasons for this specification must be reviewed.

## 11.2 IRLM

IMS multi-system data-sharing in the sysplex environment requires the use of Internal Resource Lock Manager (IRLM). If you do not currently use IRLM, you must plan for its use in the sysplex environment. IRLM is a resource-locking component of IMS that runs in its own address space in each MVS image is an IMS data-sharing environment. You need one IRLM per MVS image that contains an IMS subsystem. IRLM is mandatory for IMS record-level data-sharing. IRLM functions include the following:

- Takes over as lock manager from the program isolation lock manager.
- Maintains the integrity of data shared between multiple subsystems.

### 11.2.1 IRLM Subsystem Names

MVS is preconditioned for subsystem names IRLM and JRLM. Other names must be added by specifying them in IEFSSNxx member of SYS1.PARMLIB or by adding them dynamically with the SETSSI command. Since the effects of the SETSSI command are lost when MVS is restarted, the specification in the IEFSSNxx member should be made eventually. These subsystem names should be created according to the naming conventions for IRLM subsystem names.

### 11.2.2 IRLM as Lock Manager

This can be done before any use of block-level data-sharing is attempted. The advantage to doing this early is that it allows an installation to learn about operating the IRLM and test some operational procedures.

The local deadlock detection time should be set to resolve deadlocks quickly. The default of five seconds might be appropriate, however, some installations might want to make it even shorter. IRLM enforces a maximum deadlock time of five seconds. The global deadlock cycle is set to one regardless of what is specified.

### 11.2.3 IRLM Scope

The IRLM SCOPE parameter determines if the IRLM can work with other IRLMs.

SCOPE=LOCAL specifies that there will be only one IRLM in the data-sharing group. This limits data-sharing to one MVS system. LOCAL also implies that no lock structure will be built. Since there is no other IRLM with which to share locking information, there is no need for a lock structure.

SCOPE=GLOBAL or NODISCON specifies that there might be multiple IRLMs in the data-sharing group. They also imply that a lock structure will be built. For performance and availability reasons, the use of NODISCON instead of GLOBAL is recommended. This is explained in 18.3, “Use of IRLM SCOPE=NODISCON” on page 180.

It is reasonable to implement an IRLM with SCOPE=LOCAL as a step on the way to data-sharing. This is explained in 11.7, “Order of Implementation Steps” on page 114.

## 11.3 DBRC

RECONs, database registration, and skeletal JCL must be reviewed and updated for data-sharing.

### 11.3.1 SHARECTL

The RECONs must run in share control mode for data-sharing. Most installations will already be using this mode. It is required for IMS 6.1, even without data-sharing. It is also required for DBCTL, even without data-sharing. IMS 5.1 TM users have the option of using recovery control mode when they are not using data-sharing. These installations must change their RECONs to run in share control mode. This is done with the following command.

```
CHANGE.RECON SHARECTL
```

### 11.3.2 SHARELVL

Databases to be shared across IMS systems using different IRLMs must be registered at SHARELVL(3). SHARELVL(2) allows a database to be shared across IMS systems using the same IRLM. Typically, users of block-level data-sharing in a Parallel Sysplex want the capabilities of SHARELVL(3). Some installations use SHARELVL(2) to find potential block-level sharing lock conflicts before they implement a lock structure.

The best time to register databases at SHARELVL(3) is after the IRLM has been implemented but before sharing with another IMS subsystem is needed. The combination of the use of the IRLM and SHARELVL(2) or SHARELVL(3) causes level 3 and 4 locks to be used for KSDS block locks. This could cause locking conflicts or deadlocks to occur within one subsystem. Using SHARELVL(2) or SHARELVL(3) for one database at a time is an easy way to discover if there are potential locking problems for these databases. It is easy to revert to SHARELVL(1) if a problem exists that must be addressed. SHARELVL(1) eliminates block locks.

Database registrations can be changed individually:

```
CHANGE.DB  DBD(xxxx) SHARELVL(3)
CHANGE.DB  DBD(yyyy) SHARELVL(3)
```

Or they can all be changed, and those that are not going to be shared at the block-level can be changed back.

```
CHANGE.DB  ALL SHARELVL(3)
CHANGE.DB  DBD(zzzz) SHARELVL(1)
```

When SHARELVL(2) or SHARELVL(3) is specified for a DEDB database, structure names must be specified for the VSO areas. This is done with the CHANGE.DBDS command. The use of buffer lookaside for these areas is also specified with this command.

### 11.3.3 Skeletal JCL

Skeletal JCL for archiving and Change Accumulation should be reviewed.

Each IMS subsystem will have its own archive process; however, they will probably share the same skeletal JCL for archive. The subsystem ID (%ssid) is available to the GENJCL process and can be used to create different data set names for different IMS subsystems. The skeletal JCL supplied with IMS uses the following name for SLDSs.

```
IMS.SLDSP.%SSID.D%ARDATE.T%ARTIME.V%ARVERS
```

This provides unique names for different IMS subsystems.

Because Change Accumulation is required for database recoveries with data-sharing, Change Accumulation skeletal JCL must be in place.

## 11.4 Database Data Set Sharability

There are VSAM share options and JCL DISP= parameter requirements for sharing data sets across multiple address spaces. When these are set, database protection is provided only by DBRC and RACF. It is important that all access to the databases be done with DBRC active.

### 11.4.1 VSAM SHAREOPTION(3 3)

SHAREOPTION(3 3) is required for all VSAM full function database data sets for block-level data-sharing. CBUF processing is invoked by the combination of SHAREOPTION(3 3) and DISP=SHR. CBUF processing is required for the generation of information about the extension of data sets.

### 11.4.2 DISP=SHR in JCL or DFSMDA

The specification of DISP=SHR is required for VSAM CBUF processing and for the sharing of database data sets in general.

## 11.5 CFNAMES Statement

The CFNAMES statement is used to specify the structures that will be used by an IMS subsystem. Since all IMS systems in a data-sharing group must use the same structures, all must have the same structure names in their CFIRLM, CFVSAM, and CFOSAM specifications. The CFNAMES statement is specified in the DFSVSMxx member of PROCLIB for online systems and in the DFSVSAMP DD statement data set of batch job steps.

The CFNAMES statement must have a value specified for the CFIRLM keyword. This is the name of the lock structure that a SCOPE=GLOBAL or NODISCON IRLM will use. If there is no CFNAMES statement, a SCOPE=GLOBAL or NODISCON IRLM will use its LOCKTABL= specification for its lock structure name.

The CFNAMES statement, if included, must have a CFVSAM keyword and a CFOSAM keyword, but values for them are not required. If a value is supplied for CFVSAM, buffer invalidations for VSAM database data sets will be done using the named VSAM structure. Similarly, if a value is supplies for CFOSAM, buffer invalidations for OSAM data sets will be done using the name OSAM structure. If there is no CFNAMES statement or if a CFVSAM value is not specified, a VSAM structure will not be built. Similarly, if there is no CFNAMES statement or if a CFOSAM value is not specified, an OSAM structure will not be built. If an access method does not have a structure, its buffer invalidations are done using notifications. Notifications are messages sent between IMS subsystems using IRLM communications. This method was used for buffer invalidations in the implementation of data-sharing used by IMS releases previous to IMS 5.1.

The CFOSAM specification also is used to specify the ratio of directory entries to data elements in the OSAM structure. The values used by the IMS subsystem that builds the structure will determine this ratio. Systems that connect to the structure after it is built, can specify different values for the ratio; however, they will not affect how the structure is allocated. Only the builder determines the ratio. Since an installation can not have complete control over which IMS subsystem will always build a structure, all CFNAMES statements should have the same values for CFOSAM. This will ensure that the structure is always built with the same directory entry to data element ratio. If the directory entry and data element ratio values are not specified in the builders CFNAMES statement, IMS will calculate a default value for the ratio. Since this is not likely to be what an installation desires, the values should always be explicitly specified.

IMS initialization includes a verification that all IMS systems in the data-sharing group are using the same names for the three structures in their CFNAMES statement. If an IMS subsystem attempts to join a data-sharing group using any name that differs, its initialization will fail. Once an IMS subsystem has identified to an IRLM, these values are retained by the IRLM. The IRLM must be stopped and restarted to change these values.

## 11.6 DEDB Statements in the DFSVSMxx Member

IMS builds default buffer pools for users of VSO, but it is best to specify VSO buffer pools in the DFSVSMxx member. This is done with DEDB statements. Each DEDB statement defines a unique buffer pool. These pools are built at IMS initialization. They are not used unless VSO areas with SHARELVL(2) or SHARELVL(3) are opened.

## 11.7 Order of Implementation Steps

The following is the recommended order of implementation steps. The intention is to minimize the impact of any one step. Reasons for this order are given under each step.

It is assumed that the IRLM is not currently used. Many installations already will have implemented some of the initial steps. For example, they might have SHARECTL RECONs with databases registered at SHARELVL(1).

1. Create SHARECTL RECONs.

   This step applies only to users of IMS 5.1. IMS 6.1 does not allow the use of RECOVCTL RECONs. If any databases are not registered, it is better to begin their use of DBRC with SHARECTL. Any operational or procedural problems with SHARECTL will be discovered as databases are registered. This can be done for a small number at any time. If all databases are registered under RECOVCTL and then SHARECTL is implemented, all SHARECTL-related problems will appear simultaneously.

2. Register databases at SHARELVL(1).

   This is a typical registration level for installations not using block level data-sharing. It allows Concurrent Image Copy (CIC) to be executed and allows two data sets in the same database to be image copied simultaneously.

3. Update IMS system definition for data-sharing.

   Databases cannot be shared if they use an access intent of exclusive (EX). EX is the default value on the DATABASE macro. This change can be made at any time before a second IMS subsystem is implemented.

4. Implement SHAREOPTION(3 3) and DISP=SHR for VSAM database data sets.

   This **must** be done before the IRLM is used. The open of a VSAM database data set will fail if the IRLM is used and a SHARELVL of 1, 2, or 3 is used unless both SHAREOPTION(3 3) and DISP=SHR are specified.

   It is not necessary to specify DISP=SHR for batch jobs or utilities that do not use IRLM.

5. Implement IRLM with SCOPE=LOCAL for the online system.

   This will cause the following locking changes:

   - Segment locks for full function databases will be eliminated.
   - Deadlock detection will be done on a timer basis. This will cause involved locks to be held longer and might increase the contention for those locks.

   These could cause increased lock contention.

   The IRLM subsystem name must be known to the operating system; therefore, it must be included in the IEFSSNxx member or added by a SETSSI command before this step is taken.

6. Define the CF structures in the XCF Administrative Data Utility.

   Defining the structures will have no effect before they are specified to IMS or IRLM.

7. Include a CFNAMES statement in DFSVSMxx.

A CFNAMES statement with CFOSAM and CFVSAM specifications will cause IMS to build the buffer invalidation (cache) structures. Read and register will occur. Since data-sharing is not yet being done, no buffer invalidations will occur; however, the overhead of read-and-register operations will be added to the system.

If IMS/ESA V6.1 is being used, include the dirratio and elemratio values in the CFOSAM specification.

A value for CFIRLM must be specified. Since the IRLM is still SCOPE=LOCAL, the lock structure will not be built and no additional locking overhead will be added.

8. Include DEDB statements in DFSVSMxx for VSO areas

   DEDB statements define VSO buffer pools. These are required for VSO areas which are shared. The buffer pools are built at IMS initialization. They will not be used unless DEDBs with VSO areas are registered at SHARELVL(2) or SHARELVL(3).

9. Register databases at SHARELVL(2).

   This will cause block locks to be requested. Locks for updates to KSDS indexes might conflict within the same IMS subsystem. These can be found now.

   The databases can be registered one at a time or in groups. This will limit the effects at any time.

   Some users can skip this step. This will give them fewer steps in the implementation plan, but any potential KSDS index lock conflicts will not be found until the databases are registered at SHARELVL(3).

   DEDB VSO areas will be loaded into structures, and the VSO buffer pools will be used when the databases are registered at SHARELVL(2).

10. Implement IRLM with SCOPE=NODISCON for the online system.

    This will cause the lock structure to be built. Since no databases are registered at SHARELVL(3), only busy locks, data set reference locks, and command locks are placed in the lock structure. Block locks, database record locks, and Fast Path CI locks are not placed in the structure with SHARELVL(1) or SHARELVL(2). The overhead of this step will depend on the number of updates to KSDSs. These updates require busy locks.

11. Register databases at SHARELVL(3).

    This will cause all locks for the databases to be placed in the lock structure. No new lock conflicts will occur; however, the overhead of the CF accesses will be added to the system.

    The databases can be registered one at a time or in groups. This will limit the effects at any time.

    If step 9, the registration of databases at SHARELVL(2), was skipped. New lock conflicts might be introduced in this step. These are the potential conflicts for KSDS indexes that are explained in step 9.

12. Implement another IMS subsystem.

    This implements actual data-sharing. It might create new lock conflicts, additional lock processing, and buffer invalidations.

## 11.8 IMS Procedures for Data-Sharing

IMS procedures should be reviewed when implementing data-sharing.

The Table 2 identifies the IMS procedures (PROCs) and parameters which must be reviewed prior to migration to data-sharing. In some cases, they might need to be changed. RGSUF must be specified on the IMS or DBC procedure. Other parameters for the IMS and DBC procedures can be specified in the IMS system definition, in the IMS or DBC PROC, or in member DFSPBxxx. We recommend using DFSPBxxx.

*Table 2 (Page 1 of 3). IMS Procedures Summary*

| PROC NAME | PARAMETER | COMMENTS |
|---|---|---|
| DBBBATCH<br>DLIBATCH | All | The assumption is that IMS batch jobs currently using block-level data, data-sharing will continue to do so and that IMS batch jobs not using block-level data-sharing will not be changed to use it. Batch jobs not using data-sharing do not require changes to their execution parameters. The following parameters are for block-level data-sharing jobs. They also apply to executions of the Batch Backout utility. |
| | IRLM | This specifies that the IRLM is to be used. The value must be 'Y' for data-sharing. |
| | IRLMNM | This is the MVS subsystem name for the IRLM. The value of this parameter should be the same for each IMS. |
| | DFSVSAMP DD | This DD statement refers to a data set whose usage is the same as the DFSVSMxx member for online systems. For data-sharing, if a CFNAMES statement is included, it must be identical to all other CFNAMES statements in the data-sharing group. |
| DFSMPR | IMSID | If the control regions use the IMSGROUP parameter, this should match their IMSGROUP specification. If control regions do not use the IMSGROUP parameter, this must match the IMSID of the control region to which this message processing region will be connected. |
| | PREINIT | Verify that the preinitialization module member name (DFSINTxx) in IMS.PROCLIB has not changed. |
| | SSM | Identifies the external subsystem member of IMS.PROCLIB containing the parameters for connection to DB2. Since it identifies DB2 by its subsystem name, and since there can be multiple DB2 subsystems in the data-sharing group, this should be different for each IMS. |
| DLISAS | None | There are no parameters which must be changed for this PROC. The IMSID will be provided by the control region when it starts the separate address space: START DLISAS PARM=(DLS,IMSA). |
| DXRJPROC | IRLMNM | This is the MVS subsystem name for the IRLM. We recommend that all IRLMs in the data-sharing group use the same IRLMNM so that IMS subsystems and jobs can be moved without having to move the IRLM. |
| | IRLMID | Must must be different for each IRLM in the data-sharing group. |

*Table 2 (Page 2 of 3). IMS Procedures Summary*

| PROC NAME | PARAMETER | COMMENTS |
|---|---|---|
| | SCOPE | Must be GLOBAL for intersystem sharing. |
| | IRLMGRP | Identifies the XCF group to which this IRLM belongs. All IRLMs in the data-sharing group must specify the same XCF group. Before APAR PQ16103, this was the GROUP parameter. |
| | LOCKTABL | The parameter is not used when a CFNAMES statement is used by IMS. |
| FPUTIL | IMSID | If the control regions use the IMSGROUP parameter, this should match their IMSGROUP specification. If control regions do not use the IMSGROUP parameter, this must match the IMSID of the control region to which this Fast Path region will be connected. |
| IMS or DBC | DBRCNM | This parameter identifies the name of the DBRC PROC which is automatically started by the control region. The value of this parameter should be different for each IMS. |
| | DLINM | This parameter identifies the name of the DLISAS PROC which is automatically started by the control region. The value of this parameter should be different for each IMS. |
| | IMSID | This is the MVS subsystem name for the IMS control region. The value of this parameter must be different for each IMS. |
| | IMSGROUP | This is a generic name by which multiple control regions can be known. The specification of this parameter is optional, but recommended. When specified, all IMS systems in an IMSPlex should specify the same value. To make use of the IMSGROUP function, dependent regions must specify the same value for their IMSID parameter.<br>IMSGROUP was introduced by APAR PQ21039. |
| | IRLM | This specifies that the IRLM is to be used as the lock manager. The value of this parameter should be set to ′Y′. |
| | IRLMNM | This is the MVS subsystem name for the IRLM. The value of this parameter should be the same for each IMS. |
| | PRDR | This parameter identifies the name of the IMSRDR PROC used by /START REGION commands. |
| | RGSUF | This parameter identifies the ′xxx′ portion of the DFSPBxxx member name in IMS.PROCLIB where the rest of the parameters should be specified. |
| | SSM | Identifies the external subsystem member of IMS.PROCLIB containing the parameters for connection to DB2. Since it identifies DB2 by its subsystem name, and since there can be multiple DB2 subsystems in the data-sharing group, this should be different for each IMS. |

*Table 2 (Page 3 of 3). IMS Procedures Summary*

| PROC NAME | PARAMETER | COMMENTS |
|---|---|---|
| | SUF | This parameter identifies the 1-character suffix for the control region in IMS.RESLIB. If the IMSs are sharing one RESLIB, and are not identical clones, this parameter must be different for each IMS. |
| | VSPEC | This parameter identifies the PROCLIB member DFSVSMxx, which contains buffer pool specifications and other run time parameters. If the IMS systems are not exact clones, this parameter should be different for each IMS. The CFNAMES statement is also in DFSVSMxx. All CFNAMES statements in the data-sharing group must be identical. |
| IMSBATCH | IMSID | If the control regions use the IMSGROUP parameter, this should match their IMSGROUP specification. If control regions do not use the IMSGROUP parameter, this must match the IMSID of the control region to which this BMP region will be connected. |
| | PREINIT | Verify that the preinitialization module member name (DFSINTxx) in IMS.PROCLIB has not changed. |
| | SSM | Identifies the external subsystem member of IMS.PROCLIB containing the parameters for connection to DB2. Since it identifies DB2 by its subsystem name, and since there can be multiple DB2 subsystems in the data-sharing group, this should be different for each IMS. |
| IMSFP | IMSID | If the control regions use the IMSGROUP parameter, this should match their IMSGROUP specification. If control regions do not use the IMSGROUP parameter, this must match the IMSID of the control region to which this Fast Path region will be connected. |
| | PREINIT | Verify that the preinitialization module member name (DFSINTxx) in IMS.PROCLIB has not changed. |
| | SSM | Identifies the external subsystem member of IMS.PROCLIB containing the parameters for connection to DB2. Since it identifies DB2 by its subsystem name, and since there can be multiple DB2 subsystems in the data-sharing group, this should be different for each IMS. |

## 11.9 Data-Sharing Performance Considerations

Aside from normal performance considerations, there are some additional considerations when running in a data-sharing environment.

The analysis of data-sharing performance considerations should be based on the information in the IMS/ESA Block Level Data Sharing class (U3767).

The performance effects of data-sharing should be understood. Most effects can be predicted by an analysis of the workload and the databases. Of course, testing in a data-sharing environment can also be used to reveal performance impacts. Actions to limit these effects can include changes in database parameters, in scheduling, and in the frequency of reorganizations.

Areas to evaluate include:

- Mass deletes with KSDSs

  CI reclaim for KSDSs is not done with block-level data-sharing in IMS V5. This can lead to many I/Os when a large range of KSDS records is deleted. An investigation should be done to identify any applications that do such deletes. If possible, a reorganization or REPRO should be done after these deletes.

- High update activity to KSDSs

  Updates to a KSDS require the busy lock for the data set. This lock is held only for the duration of the update, however, it can lead to a high volume of lock requests. Inserts require a level 8 (exclusive) lock and will be single threaded with other updates.

  Updates also require a block lock on the data component CI. These are never shared across subsystems and are held until application sync point time.

  Processes which do many updates to KSDSs should be investigated for potential increased lock processing. Batch job scheduling might need to be adjusted to avoid conflicts.

- Small databases with high activity including some updates

  Small databases with high activity are likely to have lock conflicts. Updates will require block locks and cause buffer invalidations. All users will require database record locks.

  Installations should examine small databases with high activity and evaluate the potential effects of lock conflicts and buffer invalidations. It might be possible to reduce these effects by changing database parameters, such as free space and block sizes, or by caching OSAM blocks in a Coupling Facility.

- Databases with ″hot spots″

  Databases with high activity to a small number of records have performance characteristics similar to those of small databases with high activity. These databases should also be investigated for impacts of data-sharing.

- Get next calls which process many HDAM Root Anchor Points (RAPs) between root segments

  Get next processing through HDAM root segments includes locking each RAP. This is true in any locking environment. With block-level data-sharing the cost of acquiring these locks is higher since they normally require access to a coupling facility. If an HDAM database has many RAPs with few roots, one get next call can require many lock requests. This is usually addressed by reducing the number of RAPs per block.

  Installations should examine HDAM databases for the ratio of RAPs to root segments. If it is very high and get next processing of roots is done, reducing the number of RAPs will improve performance.

## 11.10 Sources of Performance Information

The following sources of information are available for use in analyzing potential performance effects of data-sharing.

- IMS System Utilities/Database Tools HD Pointer Checker

  This product has reports about databases and the spread of segments, free space, RAPs in them. The HD Tuning Statistics Report shows the total number of RAPs and the number of root segments. This is useful for finding potential problems with too many or too few RAPs.

- HD Unload Utility (DFSURGU0)

  The number of roots in the database can be obtained from the statistics reported by the HD Unload utility (DFSURGU0). This might be useful in determining which databases may have locking conflicts due to database record lock conflicts.

- DBD Source

  The number of RAPs in a database can be calculated from information in the DBD. The RMNAME parameter on the DBD statement includes the specification of the number of RAPs per block and the number of blocks in the root addressable area.

- LISTCAT reports

  The statistics section of a LISTCAT report shows the number of records updated, inserted, deleted, and read. This is useful for determining whether a VSAM data set has high update activity.

- IMS Monitor

  The IMS Monitor contains a wealth of information about accesses to databases. The database buffer pool reports are useful in determining the amount of read and write activity to databases. The program IWAIT reports show activity to individual database data sets. Unfortunately, the data is not aggregated by database or data set. Most reports aggregate by PSB or region. IMSASAP II reports most of the same data, but aggregates it by database data set.

  Turning on the IMS Monitor trace should not degrade the performance of the system. Nevertheless, using a small block size for the IMS Monitor data set, or a small number of buffers for it, might cause performance problems. This degradation can be avoided by doing the following:

  - When not using dynamic allocation for the data set, always specify the BLKSIZE subparameter on the DD statement DCB parameter. For DASD, use half-track blocking. This is BLKSIZE=27992 for 3390s. For tape, use a 32K block size. This is BLKSIZE=32768.

    This specification must be made. When using an old data set without this specification, IMS will change the block size to the default. The default block size is calculated by IMS and will generally be from 1048 to 4096 bytes. This is too small for most users.

  - When using dynamic allocation for a tape data set, always specify BLKSIZE in the DFSMDA dynamic allocation statement. Use a 32K block size. This is BLKSIZE=32768.

  - When using dynamic allocation for a DASD data set, always specify BLKSIZE in the DFSMDA dynamic allocation statement. Use half-track blocking. This is BLKSIZE=27992 for 3390s. See IMS V6 APAR PQ04670/PTF UQ05808, which allows dynamic allocation of a preallocated monitor DASD data set (DISP=OLD).

– Always specify the BUFNO parameter on the DD statement or on the DFSMDA dynamic allocation statement. Make the value at least 5. For systems doing over 100 transactions per second or with more than 10 concurrently executing BMPs, specify a larger value. BUFNO=20 should be adequate for any system.

For example, one could use the following DD statement:

```
//IMSMON DD DSN=IMS.IMSMON,DCB=(BLKSIZE=27992,BUFNO=10),DISP=SHR
```

IMS Monitor buffers reside in ECSA. The use of 10 buffers with a 27992 block size would require less than 280 KB of ECSA.

- IMSASAP II

  The Total System IWAIT Detail Report of the IMSASAP II product (5798-CHJ) lists all of the database data set DDNAMEs and the number of IWAITs for each during a monitoring period. It is useful for determining which database data sets have the most I/O activity.

The following sources of information are available for use in analyzing performance effects when data-sharing is implemented.

- RMF

  Coupling Facility use is reported in the RMF Coupling Facility Activity reports. The number of accesses to each structure, the average service times for CF requests, the number of real and false lock contentions, and the number of buffers invalidated by cross invalidations are among the most useful data in these reports.

- IMS Monitor

  The IMS Monitor reports waits for locks in its Program I/O reports. The database data set for which the lock request was made is reported by ″PI ddname....″ in the DDN/FUNC column.

We recommend keeping historical performance information, including information about the performance of the system before data-sharing is implemented. Usually, the best information is IMS Monitor reports from peak times. This should include both online transactions and batch (BMP) jobs. The effects of data-sharing can be determined by contrasting reports from before and after the implementation of data sharing.

## 11.11 Data Sharing Coupling Facility Performance Impacts

The primary overhead from implementing data-sharing in a Parallel Sysplex environment comes from accessing Coupling Facilities. Different steps can introduce increases in the number of CF accesses. This section explains what factors influence the use of the CF by IMS and the IRLM.

The most significant increases in CPU utilization from CF accesses typically occur with the following steps.

- Use of IRLM with CFNAMES statement in IMS

Specification of CFOSAM and CFVSAM values in CFNAMES causes databases with SHARELVLs of 1 or higher to use these structures.

- Use of IRLM with SCOPE=GLOBAL or NODISCON

  Updates to index databases registered at SHARELVLs of 1 or higher cause accesses to the IRLM lock structure. The accesses are for busy locks.

- Use of IRLM with SCOPE=GLOBAL or NODISCON and databases registered at SHARELVL(3)

  All locks for these databases are placed in the IRLM lock structure.

More complete explanations of which specifications cause CF accesses follow.

## 11.11.1 IRLM CF Access

The IRLM accesses its lock structure for lock and unlock requests. The type of lock and the SHARELVL for the database determine if the lock is placed in the structure.

A lock structure is built when the SCOPE=GLOBAL or when NODISCON is specified for the IRLM. If SCOPE=LOCAL is used, there is no lock structure, and the IRLM does not access the CF. If SCOPE=GLOBAL or NODISCON is used, CF accesses are required for some locks. The following sections explain when the different types of locks are placed in the lock structure.

### 11.11.1.1 Busy Locks

Each full-function database data set has a busy lock. These locks are requested for the following activities:

- Data set open
- Data set close
- New block creation in data set
- Update to KSDS

These locks are placed in the CF for all databases with SHARELVL of 1 or higher. The locks are released when the operation completes. CF accesses for these locks might be significant for KSDSs. Heavily updated HIDAM indexes and secondary indexes might require many lock requests. They are the only significant use of the CF by locks for databases with SHARELVLs less than 3.

### 11.11.1.2 Database Record Locks

Database record locks are requested when a full-function database record is accessed. These locks are requested for all SHARELVLs; however, only locks for databases with SHARELVL of 3 are placed in the lock structure. These locks are typically a large part of the accesses to a lock structure.

### 11.11.1.3 Block Locks

Block locks are requested when a full function database OSAM block or VSAM CI is updated. These locks are requested for databases with SHARELVL of 2 or 3; however, only locks for databases with SHARELVL of 3 are placed in the lock structure. These locks are typically a large part of the accesses to a lock structure.

#### 11.11.1.4 Fast Path DEDB CI Locks

CI locks are requested when DEDB CIs are read. These locks are requested for all SHARELVLs, but only those for databases with a SHARELVL of 3 are placed in the CF.

When VSO data-sharing is used, CI locks are requested when an application program acquires a CI. It can be acquired by reading it from DASD, by reading it from a cache structure, or by finding it in the area′s look-aside pool. For preloaded VSO areas, a CI lock is also requested for each CI read during the preload process.

#### 11.11.1.5 Fast Path DEDB UOW Locks

UOW locks are requested when DEDB UOWs are read by a utility or HSSP process. They are also requested by other applications using an area when an HSSP or utility process is active for the area. These locks are requested for all SHARELVLs, but only those for databases with a SHARELVL of 3 are placed in the CF.

#### 11.11.1.6 Other Locks

There are several other types of locks that are requested by IMS in a data-sharing environment. These include command locks and data set reference locks. The number of requests for these locks is typically a very small percentage of the total lock requests. Thus, they rarely, if ever, have an important effect on data-sharing overhead.

#### 11.11.1.7 Processing Unlock Requests

When unlock requests are processed by the IRLM, it must access the CF to release locks in the lock structure. IMS can release one or more locks with one unlock request. In any case, the IRLM makes only one access to the CF to process up to 128 locks in an unlock request. If more than 128 locks are released in one unlock request, a CF access for each 128 locks is required. An example of unlocking multiple locks is the unlock request made by IMS when an application program reaches a sync point. It releases all of its locks with one request.

### 11.11.2 IMS CF Access

IMS accesses the CF for shared full function databases and shared shared DEDB VSO areas.

#### 11.11.2.1 Full-Function VSAM

The VSAM cache structure is accessed for VSAM database data sets when the following conditions are met.

- An IRLM is used by the IMS subsystem.
- A CFNAMES statement is used with a value for CFVSAM specified.
- The database is registered at SHARELVL 1, 2, or 3.

An access is made to the structure for each read and each write to DASD for the database data set.

### 11.11.2.2 Full-Function OSAM

The OSAM cache structure is accessed for OSAM database data sets when the following conditions are met.

- An IRLM is used by the IMS subsystem.
- A CFNAMES statement is used with a value for CFOSAM specified.
- The database is registered at SHARELVL 1, 2, or 3.

An access is made to the structure for each read and each write to DASD for the database data set. When OSAM caching is used, there might be additional accesses to the CF. Data read from the CF structure requires a CF access. Usually, only one CF access is required. The exception is for block sizes greater than 4 KB using OSAM pools that cache only changed data. Reads of these blocks from the CF use two CF accesses. When data is read from the CF, the accesses typically have longer service times producing greater overhead. Of course, if this eliminates a read from DASD, the overall effect can be to reduce CPU utilization.

### 11.11.2.3 DEDB VSO

DEDB VSO area structures are accessed for DEDBs when the following conditions are met.

- An IRLM is used by the IMS subsystem.
- The DEDB area is defined to DBRC with one or two structures for data-sharing.

If two structures are used, all CF writes are to both structures. CF reads are only from the primary structure unless a failure occurs. CF writes occur when any CI in the area is updated. CF read requests are done when a CI is not found in the DEDB private buffer pool for the area.

When the preload option is used for a shared area, the first IMS subsystem to connect to the structure writes each CI to the structure when the area is opened.

Cast-out processing requires accesses to the CF. Cast-outs are done at each system checkpoint and at area close time by each sharing IMS subsystem. An access to the CF is required for each changed CI that is cast out. When two structures are used, an access to the second structure is also required for each cast-out CI. A small number of additional accesses are required to determine which CIs should be cast out and to handle the serialization of cast-out processing.

# Chapter 12. Shared Queues Enablement

**TM Only**

This chapter applies only to IMS TM.

The following tasks are required to enable shared queues in the target environment. Shared queues are invoked when the IMS procedure includes a specification for SHAREDQ.

## 12.1 Program Properties Table

The MVS program properties table must be updated to include CQS. Member SCHEDxx of SYS1.PARMLIB must include an entry for CQSINIT0. CQS cannot be started without this entry.

## 12.2 Structure Definitions

The structures used by CQS must be defined in a CFRM policy. These include the primary and overflow structures for the full-function queues, the primary and overflow structures for EMH, and the logger structures used by both full-function and EMH.

## 12.3 Log Stream Definitions

The log stream definitions must be included in the logger (LOGR) policy. The log stream definitions include specifications of the structures to be used for the log streams, the size of logger buffers, the offload parameters, the use of duplexing, the high-level qualifier for logger data set names, and the SMS parameters used for the log stream data sets.

## 12.4 System Checkpoint Data Sets

Each CQS address space that is connected to a shared queues structure maintains a system checkpoint data set for each structure pair. Neither CQS address spaces nor shared queues structures share the system checkpoint data sets. The CQS initialization process dynamically allocates system checkpoint data sets. These data sets are VSAM ESDSs. They are defined with the MVS/DFP `DEFINE CLUSTER` command and a small space allocation (for example, `TRK(1 1)` is sufficient), function messages, and another for EMH. These data sets are VSAM ESDSs. They are defined with the MVS/DFP `DEFINE CLUSTER` command.

## 12.5 Structure Recovery Data Sets

The structure recovery data sets must be defined. All CQSs using a pair of primary and overflow structures, share one pair of structure recovery data sets. There is one pair for full-function messages and another pair for EMH. These data sets are VSAM ESDSs. They are defined with the `MVS/DFP DEFINE CLUSTER` command. Space allocated to these data sets should be slightly greater than the structure SIZE defined for a structure pair in the CFRM policy.

## 12.6 CQS Procedure

The CQS procedure must be created. It executes program CQSINIT0. Parameters include PROCLIB members used for execution. Other parameters can be used to override specifications in these members, such as the CQS subsystem name (SSN), the CQS group name, and the use of ARM. The PROCLIB members referenced are CQSIPxxx, CQSSGxxx, and CQSSLxxx.

## 12.7 BPE Configuration PROCLIB Member

The BPE configuration PROCLIB member must be created. It defines configuration parameters for BPE. BPE and CQS traces are specified here.

## 12.8 CQS Initialization Parameters PROCLIB Member (CQSIPxxx)

The CQS Initialization Parameters PROCLIB member (CQSIPxxx) must be coded. It contains parameters to be used for this execution on CQS. They include the the CQS subsystem name (SSN), the CQS group name, and the use of ARM. The CQS Local and Global Structure Definition PROCLIB members also can be specified.

## 12.9 CQS Local Structure Definition PROCLIB Member (CQSSLxxx)

The CQS Local Structure Definition PROCLIB member (CQSSLxxx) must be coded. It contains parameters indicating the names used for the primary message queues structure and the primary EMH structure. The system checkpoint data sets associated with these structures are also specified here. The system checkpoint log record counts can be specified. Since each CQS uses its own system checkpoint data set, each CQS must use a different member.

## 12.10 CQS Global Structure Definition PROCLIB Member (CQSSGxxx)

The CQS Global Structure Definition PROCLIB member (CQSSGxx) must be coded. All CQSs sharing a set of structures must have identical values specified for the parameters in this member. This is done most easily by having them all point to the same member. The parameters include the names of the structures, the structure recovery data sets, and the log streams for both full-function and EMH messages.

## 12.11 Security for CQS Structures

If access security for shared queues structures is desired, it must be defined in the security product. There are two ways to control access to CQS shared queues structures with a security product, such as RACF. First, access by a CQS client can be controlled using profile name of CQSSTR.structure-name, where structure-name is the name of the primary structure. Second, access by any user of Cross-System Extended Services (XES) can be controlled by using a profile name of IXLSTR.structure-name, where structure-name is the name of the structure. Since CQS clients use XES to access structures, the second way is recommended. It protects the structure from all potential users, not just CQS clients. The use of IXLSTR.structure-name profiles is described in 16.2, “Security” on page 160.

## 12.12 ARM Policy Updates

The ARM policy might need to be updated to include CQS restart information. Each CQS should belong to the same restart group as its partner IMS.

## 12.13 IMS Shared Queues PROCLIB Member (DFSSQxxx)

The DFSSQxxx member of PROCLIB must be created. It specifies shared queues parameters to IMS. These include the CQS execution procedure member name, the CQS subsystem name, the shared queues group name, the names of the primary message queue structures, and the parameter indicating if IMS is to wait for rebuilds of the EMH structure.

## 12.14 QMGR and Shared Queues Traces

QMGR and shared queue traces can be set in the DFSVSMxx member of PROCLIB. If they are not set on in this member, they can be turned on at execution time by the /TRACE command.

## 12.15 IMS Procedure

The IMS start up parameters must be modified. Shared queues support for IMS is invoked when the SHAREDQ parameter in the IMS DFSPBxxx member is specified. It indicates the suffix of the DFSSQxxx member of PROCLIB.

## 12.16 IMS Procedures for Shared Queues

IMS procedures should be reviewed when implementing shared queues.

Shared queues are an execution time option. If the SHAREDQ parameter is specified with a non-null value for the IMS or DCC procedures, shared queues are used. When shared queues are used, some parameters in the IMS and DCC procedures are not used, and some have different meanings. Table 3 on page 128 identifies the message queue parameters in the IMS and DCC procedures. It lists the IMS PROCs and parameters which should be reviewed prior to migration. In some cases, they might need to be changed. These parameters can be specified in the IMS or DCC procedures or in member DFSPBxxx. We recommend using DFSPBxxx.

*Table 3. IMS Shared Queues Procedure Parameters*

| PROC NAME | PARAMETER | COMMENTS |
|---|---|---|
| IMS or DCC | SHAREDQ | The specification of any value for this parameter causes shared queues to be used. Otherwise, traditional queuing is used. The value specifies a three-character suffix for the shared queues PROCLIB member, DFSSQxxx. |
| | QBUF | This parameter is used by both shared queues and traditional queuing. It specifies the number of message queue buffers. For shared queues, this is an initial number of buffers, which can be dynamically increased. |
| | EXVR | This parameter is used by both shared queues and traditional queuing. It determines whether or not the message queue buffers are long term page fixed. |
| | QBUFSZ | This parameter is used only by shared queues. It is ignored with traditional queuing. It specifies the size of the message queue buffers. |
| | SHMSGSZ | This parameter is used only by shared queues. It is ignored with traditional queuing. It specifies the short message logical record length. |
| | LGMSGSZ | This parameter is used only by shared queues. It is ignored with traditional queuing. It specifies the long message logical record length. |
| | QBUFHITH, QBUFLWTH, QBUFPCTX, and QBUFMAX | These parameters are used only by shared queues. They are ignored with traditional queuing. They are used to control the dynamic expansion of the number of buffers in the message queue pool. The initial number is set by the QBUF parameter. |
| | QTU and QTL | These parameters are used only with traditional queuing. They are ignored with shared queues. They control the issuing of messages when the use of message queue data sets crosses the specified thresholds. |

# Part 5.  MVS Parallel Sysplex Conderations

# Chapter 13. VTAM Generic Resources Enablement

**TM Only**

This chapter applies only to IMS TM.

The following tasks are required to enable generic resources in the target environment.

## 13.1 VTAM Requirements

A VTAM generic resource structure must be defined.

Generic resources is supported by VTAM when a generic resources structure is defined in the CFRM policy. The default name for this structure is ISTGENERIC. Users of VTAM 4.3 or later releases may specify a different name. This is done in the STRGR VTAM start option parameter. The name must begin with ″IST″. One structure is used by VTAM to support all generic resource groups. All of VTAM′s sharing generic resource information must specify the same structure name.

Generic resources requires VTAM 4.2 or later releases. Support for generic resources with APPC requires VTAM 4.4 or later releases and OS/390 1.3 or later releases. Generic resources also requires that IMS is executing on an APPN node.

Installations may code a Generic Resource Resolution Exit (ISTEXCGR). This exit has the capability to override the node selected by VTAM generic resources for a session establishment request.

## 13.2 IMS Requirements

The generic resource group name must be specified to the IMS systems.

IMS uses VTAM generic resources when the GRSNAME IMS execution parameter is specified. This is the generic name. Of course, all IMS systems in a generic resource group must specify the same name.

If PQ18590 is applied, the GRAFFIN parameter may be specified. This determines whether IMS or VTAM will manage generic resource affinities. GRAFFIN=IMS is the default. LU 6.1 (ISC) affinities are always managed by IMS.

## 13.3 APPC/IMS Requirements

For APPC/IMS to use generic resources, the APPC/IMS generic resource group name must be specified for each APPC/IMS system.

APPC/IMS uses VTAM generic resources when the GRNAME parameter on the LUADD statement in the APPCPMxx PARMLIB member is specified. This name must be different from the name used for the IMS generic resource group. Of course, all APPC/IMS nodes in the same generic resource group must specify the same name.

# Chapter 14. Automatic Restart Manager (ARM)

The Automatic Restart Manager (ARM) is a facility of MVS introduced in MVS/ESA SP 5.2. It can be used to automatically restart failed IMS online subsystems and certain other jobs and tasks in a Parallel Sysplex.

In an IMS data-sharing environment, it is important to restart failed IMS subsystems as quickly as possible. Failed subsystems may hold locks protecting updates. Requests for these locks result in lock reject conditions, which usually cause abends of application programs. Emergency restart releases these locks when it backs out in-flight work. These backouts eliminate any further lock rejects. ARM may be used to invoke emergency restarts more quickly and, thus, provide greater availability.

ARM distinguishes between abends of a job or task and failures of an entire MVS system. If a job or task fails, ARM may be used to automatically restart it on the same system. If a system fails, ARM may be used to restart the work on other systems in the Sysplex.

Only those jobs or started tasks which register to ARM are eligible to be restarted by ARM. IMS control regions, IRLMs, FDBR, and CQS may to do this registration. For control regions, FDBR, and CQS, it is optional. For the IRLM, it always occurs when ARM is active. IMS dependent regions (MPR, BMP, and IFP), IMS batch jobs (DLI and DBB), IMS utilities, and the online DBRC and DL/I SAS regions do not register to ARM. There is no need for ARM to restart online DBRC and DL/I SAS regions since they are started internally by the control region.

ARM is optional. If used, it must be defined in an ARM policy, and the policy must be activated. The policy is defined using the MVS Administrative Data Utility. It is activated by a SETXCF command.

ARM support in IMS was implemented through APAR PN71392 for IMS/ESA 5.1. With this support, IMS defaults to the use of ARM when run under an MVS using ARM. IMS regions that support ARM have an ARMRST parameter. If ARMRST=N is specified, the region does not register with ARM.

When ARM restarts an IMS online system, IMS will always use AUTO=Y. This is true even if AUTO=N is specified. The use of AUTO=Y eliminates the need for the operator to enter the /ERE command.

When ARM restarts an IMS online system, IMS specifies the OVERRIDE parameter when appropriate. This eliminates the requirement for an operator action during automatic restarts.

## 14.1 Exceptions to Automated Restarts of IMS

An IMS online system will not be automatically restarted by ARM in the following situations:

- An ARM policy is not active.
- There is no available system for the restart.
- ARMRST=N is specified for the IMS control region.

- IMS is normally terminated. This is done by a /CHE PURGE, FREEZE, or DUMPQ command.
- IMS is terminated by an MVS cancel unless ARMRESTART is specified on the CANCEL or FORCE command.
- Any of the following IMS abends occur:
  - U0020 - MODIFY
  - U0028 - /CHE ABDUMP
  - U0604 - /SWITCH SYSTEM
  - U0758 - QUEUES FULL
  - U0759 - QUEUE I/O ERROR
  - U2476 - CICS TAKEOVER
- IMS abends before it completes restart processing. This is used to avoid recursive abends since another restart would, presumably, also abend.

## 14.2 Restart Conditions

The ARM policy controls under which conditions a job or task is restarted. An installation may choose to have the job or task restarted on both abends and system failures or only on abends. The TERMTYPE parameter is used for this specification. TERMTYPE(ALLTERM) specifies that restarts will occur for either abends or system failures. TERMTYPE(ELEMTERM) specifies that restarts will occur only after abends. ALLTERM is the default.

## 14.3 Restart Groups

An installation defines jobs or tasks in a restart group. They are called elements. When a system failure occurs, ARM restarts all elements in a restart group on the same MVS. Of course, only those elements for which TERMTYPE(ALLTERM) has been specified will be restarted. A restart group might be composed of an IMS DBCTL control region and all of the CICSs which attach to it. Another example is an IMS DB/DC control region and the DB2 to which it attaches.

Restart groups are defined in the ARM policy for the Administrative Data Utility. The policy identifies which elements are in a restart group, where they may be restarted, and the command or JCL used to restart them.

### 14.3.1.1 Specifying the Restart Method

ARM provides three choices for method used to restart an element. They are:

- Use the same JCL that previously started the element.
- Restart the element as a batch job using JCL in a data set specified in the policy.
- Restart the element by issuing the command that is specified in the policy.

The RESTART_METHOD parameter for the element is used to specify the choice. For most installations, using the same JCL will be appropriate. This is the PERSIST specification.

### 14.3.1.2 A Restart Group Example

The following is an example of statements used to define the elements in a restart group and the MVS systems on which they may execute. PD11 is an IMS DBCTL system. PA11, PA12, and PA13 are CICS AORs that use PD11. PT11 and PT12 are CICS TORs which use these CICS AORs. This installation has named its restart groups using the MVS system on which the group normally runs and an identification of the data-sharing group. The MVS system is SYSA and the data-sharing group is IMSP. This leads to name SYSA_IMSP for the restart group.

The restart group includes an IRLM; however, it is only restarted after abends. It is not moved with the group when its system fails.

```
RESTART_ORDER
 LEVEL(1)
  ELEMENT_NAME(PD11)              IMS
 LEVEL(2)
  ELEMENT_NAME(PA1*)              CICS AORS
 LEVEL(3)
  ELEMENT_NAME(PT1*)              CICS TORS
RESTART_GROUP(SYSA_IMSP)
 TARGET_SYSTEM(SYSA,SYSB,SYSC,SYSD)
  ELEMENT(PD11)                   IMS
   RESTART_METHOD(BOTH,PERSIST)
  ELEMENT(PA1*)                   CICS AORS
   RESTART_METHOD(BOTH,PERSIST)
  ELEMENT(PT1*)                   CICS TORS
   RESTART_METHOD(BOTH,PERSIST)
  ELEMENT(PI01)                   IRLM
   TERMTYPE(ELEMTERM)
   RESTART_METHOD(BOTH,PERSIST)
```

If SYSA fails, PD11, PA11, PA12, and PA13 will be restarted on either SYSB, SYSC, or SYSD. They will be restarted using the same JCL that was used for the failed execution. PD11 will be restarted before any of the CICS AORs, and the AORs will be restarted before the TORs.

This installation has an IRLM with the same name executing on each of the MVS systems. It does not need to restart the IRLM on MVS failures. It has specified TERMTYPE(ELEMTERM) to avoid these restarts. Other installations may have other circumstances. If a group may be restarted on an MVS where an IRLM is not already executing, a specification of TERMTYPE(ALLTERM) for the IRLM may be used to restart the IRLM also.

## 14.4 Other ARM Capabilities

ARM has options to control the order in which elements are started, the number of attempts made per element, the timing of these attempts, and other actions taken upon failures. A complete description of ARM policy specifications, including these options, appears in *OS/390 Setting Up a Sysplex*, GC28-1779.

#### 14.4.1.1 Restarting Dependent Regions

ARM will not restart IMS dependent regions (MPRs, BMPs, and IFPs). These must be started by some other means. Other automated operations facilities, such as NetView or IMS Time-Control Option (TCO) may be used.

## 14.5 ARM with IRLM

IRLM added ARM support via APAR PQ06465. If an ARM policy is active, IRLM always registers once this maintenance is applied.

When using ARM to restart IRLM, module DXRRL0F1 must be in the MVS linklist. If it is not, restarts of IRLM by ARM will fail.

### 14.5.1 Restarting after IRLM Abends

If an IRLM abends, ARM will restart it on the same MVS system. In the meantime, IMS will quiesce. This includes terminating any in-flight units of work and discontinuing scheduling. When this completes, IMS will attempt to reconnect to its IRLM. Since restarting the IRLM is a very quick activity, ARM will have restarted the IRLM before IMS attempts the reconnection. So, the reconnection of IMS to its IRLM will be automatic.

### 14.5.2 Restarting after System Failures

If the same IRLM name is used by all IRLMs in a data-sharing group, restarting IRLMs for system failures is not required. The installation may always have an IRLM executing on each target system for the restart group. This will ensure that an IRLM with the correct name is available for any IMS control region restarts. In this case, the ARM policy should include a TERMTYPE(ELEMTERM) parameter for the IRLMs. This will cause ARM to restart the IRLMs only on abends.

If different IRLM names are used by IRLMs in a data-sharing group, an installation should place each IMS and its IRLM in a restart group. This will ensure that they are restarted on the same MVS. Of course, each pair of IMS and IRLM systems would have its own restart group. In this case, the ARM policy should include a TERMTYPE(ALLTERM) parameter for each IRLM. This will cause ARM to restart the IRLMs on both abends and system failures.

## 14.6 ARM with CQS

CQS includes ARM support. If ARM is active and ARMRST=N is not specified in the CQS procedure, CQS registers with ARM. ARM will restart it if CQS abends or if its MVS system fails. There are some exceptions to ARM restarts after CQS abends. Module CQSARM10 contains a table of CQS abends for which ARM restarts will not be done. Users may modify this table. The table is shipped with the following abends.

- 0001 - ABEND during CQS initialization
- 0010 - ABEND during CQS initialization
- 0014 - ABEND during CQS initialization
- 0018 - ABEND during CQS restart
- 0020 - ABEND during CQS restart

Since a CQS must execute with its IMS system, it should be included in a restart group with its IMS.

## 14.7 ARM with FDBR

The use of ARM with FDBR is explained in 19.6, “ARM and FDBR” on page 192.

## 14.8 Information for ARM Policies

When defining an ARM policy, the ARM element name and ARM element type are often required. Table 4 lists the element names and types for the various IMS components.

*Table 4. ARM Element Terms*

| **Region** | **Element Type** | **Element Name** |
|---|---|---|
| IMS Control Region | SYSIMS | IMSID |
| IRLM SCOPE=LOCAL | SYSIRLM | IRLM name concatenated with IRLM ID |
| IRLM SCOPE=GLOBAL | SYSIRLM | IRLM Group Name concatenated with IRLM name and IRLM ID |
| CQS | SYSIMS | CQSID which is ″CQS″ concatenated with the CQS subsystem name |
| FDBR | SYSIMS | IMSID of the FDBR region |

# Chapter 15. Coupling Facility

Coupling Facility structures must be implemented to support IMS block-level data-sharing and shared queues. These are the IRLM lock structure, the OSAM and VSAM cache structures, the DEDB VS cache structures, the shared queue list structures, and the MVS logger list structures used by shared queues. The sizes and placement of these structures must be determined. These sizes and placement are specified in policies which are defined using the XCF Administrative Data Utility (IXCMIAPU). The definition of the structures is also using the Administrative Data Utility. The utility is documented in *OS/390 Setting Up a Sysplex*, GC28-1779.

Recovery procedures for failures of these structures, for their CFs, and for connections to their CFs must be developed. Recovery procedures for Coupling Facility structure failures and Coupling Facility connection failures must be established. Information on these procedures is included in Chapter 20, “Recovery Procedures” on page 197.

REBUILDPERCENT values for structures will affect the automatic rebuilding of them on connection failures. The values chosen will be based on the desired configuration after one of these types of failures.

## 15.1 Coupling Facility Planning Guidelines

These guidelines are intended to provide some *rules of thumb* for initial CF structure planning.

### 15.1.1 Structure Placement Rules

The following rules govern the selection of a CF for a structure.

The system allocates the structure in the Coupling Facility in the preference list that meets the following allocation criteria, which is listed in order of relative importance from most important to least important. Volatility and failure-independence are explained below.

1. Has connectivity to the local system trying to allocate the structure
2. Has a coupling facility operational level (CFLEVEL) equal to or greater than the requested CFLEVEL
3. Has space available that is greater than or equal to the requested structure size
4. Meets the volatility requirement requested by the connector
5. Meets the failure-independence requirement requested by the connector
6. Does not contain a structure in this structure′s exclusion list.

If there is no Coupling Facility in the preference list that meets all these allocation criteria, the system determines the Coupling Facility that most closely meets the criteria. To do this, the system uses a weighting system for each of the Coupling Facilities in the structure′s preference list. The weights correspond to the list of criteria, with system connectivity having the highest weight, CFLEVEL the next higher weight, and so on down the list.

Using these weights, the system orders the Coupling Facilities that meet all requirements and attempts to allocate the structure. If two or more Coupling Facilities are assigned identical weights, the selection is made based on the order in which the Coupling Facilities are defined in the CFRM policy preference list. If the attempt to allocate the structure is not successful, the system reorders the Coupling Facilities in the preference list, ignoring the exclusion list requirement, and again attempts to allocate the structure. The system continues its allocation attempt in successively lower-weighted Coupling Facilities until allocation is successful. The system will choose the Coupling Facility that most closely meets the requirements of the connect request. If no Coupling Facility meets the allocation requirements, the connection request fails.

#### 15.1.1.1 Volatility

Connectors can request a non-volatile CF. This is a CF which maintains its storage when a power failure occurs. Typically, this means the CF has a battery back up. The IRLM and IMS do not request a non-volatile CF for their structures.

#### 15.1.1.2 Failure-independence

Failure-independence means the CF is not in an LPAR on the same machine (CEC) as the MVS system in which the connector resides. It attempts to limit the possibility of a double failure. A double failure would occur if a structure and its connector were lost simultaneously.

A connector requests non-volatility and failure-independence as a pair. That is, it requests either both of them or neither of them. Since the IRLM and IMS do not request non-volatile CFs, they also do not request failure-independence for their structures.

### 15.1.2 Initial Structure Placement

Structure placement is essentially the process of satisfying the demands of the CF users with the resources available.

1. The sum of the structure sizes on a CF must be less than the memory available on the CF.
2. Don′t configure any CF so that a single CF failure causes a catastrophic outage. It is far better to run degraded than not run at all. Make certain that there is sufficient storage and processing capacity in the remaining CFs to allow the exploiters to rebuild their structures somewhere else and continue running.
3. Structures cannot be split across CFs. The structure exists in one and only one CF at any time.

IMS does not force the placement of any structures on the same or different CFs. The factors affecting structure placement are:

- Storage on the CFs

  Obviously, the storage of a CF most be adequate to support the sum of the sizes of the structures placed on the CF. This is rule 1 listed above.

- Performance

  Performance will be affected by the ability of the CFs to satisfy requests for their services. The busiest structures may be separated on different CFs to balance the use of the CFs. This should be done for all of the structures in the Parallel Sysplex. Which structures are the busiest will depend upon the workload; however, the IRLM lock structure will typically be among the busiest structures in an installation.

- Recovery in case of failures

  A pair of DEDB VSO structures should be placed on separate CFs for recovery purposes. If they are not, a failure of one CF could cause the unavailability of the area.

### 15.1.3 Structure Sizing

Structures include such entities as fixed controls, data elements, and directory entries. The size of a structure can be calculated from the numbers of such entities. The *S/390 PR/SM Planning Guide*, GA22-7236, includes information for calculating structure sizes. Most users do not find it necessary to make these calculations. Instead, they begin with a simpler estimate and monitor the structure after it is built. The following sections give these simpler estimates.

### 15.1.4 IMS Database Manager

IMS uses two separate cache structures for full-function databases, one for OSAM buffers and one for VSAM buffers. For IMS/ESA 5.1, these cache structures contain no data and are used to maintain local buffer coherency. For IMS/ESA 6.1, the OSAM structure may also contain data.

IMS uses one or two structures for each shared DEDB VSO area. Information for estimating these structure sizes is presented in the following sections.

#### 15.1.4.1 VSAM Structures

This section applies to cache structures used by VSAM for buffer invalidations. Each unique VSAM CI that resides in a buffer pool must have an entry in the structure. Each entry is approximately 200 bytes. Approximately 50 KB bytes are required for fixed controls. If we assume that each buffer contains a different block or CI, the structure sizes can be calculated with the following formula.

```
VSAM Structure Size = (200 x # VSAM Buffers) + 50 KB
```

The number of buffers should include those used by VSAM Hiperspace.

#### 15.1.4.2 OSAM Structures

IMS/ESA 6.1 implements caching for OSAM structures. The use of caching is optional. The option is by subpool and is specified in the DFSVSMxx member for online systems and the DFSVSAMP data set for batch jobs. Caching requires data elements in the cache structure. As of APAR PQ13274, the creation of data elements is optional. Before this APAR, IMS/ESA 6.1 always built data elements even though caching was not to be used.

The format of the CFNAMES statement is:

```
CFNAMES,CFIRLM=irlmn,CFVSAM=vsamn,CFOSAM=(osamn,dirratio,elemratio)
```

Where:

`irlmn` The name of the IRLM lock structure.

`vsamn` The name of the VSAM buffer invalidation (cache) structure.

`osamn` The name of the OSAM cache structure.

`dirratio` The directory entry part of the directory entry to data element ratio. This is a 1-3 digit integer number. The minimum value is 1. The default value as of APAR PQ16250 is 999.

`elemratio` The data element part of the directory entry to data element ratio. This is a 1-3 digit integer number. The minimum value is 0. The default value as of APAR PQ13274 is 1.

The ratio of *dirratio* to *elemratio* is important, not the absolute value of either one. For example, values of 3 and 1 have the same result as values 300 and 100.

Users who do not wish to cache OSAM data should specify 0 for the elemratio parameter in the CFNAMES statement. This causes the structure to be built without data elements. When a structure is built without data elements, requests to cache data are not honored. That is, the specification of caching for a subpool will be ignored if the structure has no data elements. APAR PQ16250 added the capability to specify 0 for the elemratio parameter.

The structure storage is divided between directory entries and data elements. The directory entries use approximately 200 bytes each. The data elements are 2 KB bytes each.

Directory entries keep track of OSAM blocks that are either in a buffer pool, in the structure, or both. There must be a directory entry for each OSAM block that is in an OSAM buffer pool. There also must be a directory entry for each block that is in the cache structure but not in a buffer pool. If a block is in one or more buffer pools and also in the structure, it requires only one directory entry.

For each block that is stored in the structure, there must be enough data elements to hold it. Blocks of up to 2 KB bytes require one data element. Blocks greater than 2 KB bytes and up to 4 KB bytes require two data elements, and so forth.

To calculate the size of the structure and the correct specification for the CFOSAM parameter on the CFNAMES statements, one must know the number of unique blocks that will be tracked in the structure and the number of blocks that one wants cached in the structure at any time. You must also know the average number of data elements required to hold a cached block.

The following can be used to calculate the structure size.

```
  Structure Size = 200(A + B) + (A x (2K x C)) + D
```

Where:

A Number of cached blocks.

B Number of uncached blocks that will be in the OSAM buffer pools at any time.

C Average number of 2 KB buffers required to hold a cached block. If caching is not used, this is 0.

D 50 KB if data elements are not used, 70 KB if data elements are used.

The *dirratio* and *elemratio* subparameters of the CFOSAM parameter can be calculated as follows.

```
If C=0,
  elemratio = 0
  dirratio = any valid value

If C>0,
  dirratio = A + B
  elemratio = A x C
```

Where A, B, and C are explained above. If either ″ratio″ number is greater than three digits, they must both be reduced to keep their ratio the same. For example, if dirratio is 50,000 and elemratio is 10,000, they can be reduced to 5 and 1.

### 15.1.4.3 Sample Calculation

The following sample illustrates the calculations for specifying an OSAM structure with caching. Assume we have two sharing systems with the following characteristics:

- Number of cached OSAM blocks = 1,000

  This is the number of blocks that we wish to keep cached in the structure at any time.

- Number of OSAM buffers = 100,000

  This is 50,000 buffers in each of two systems.

- Number of uncached OSAM blocks in OSAM buffer pools = 90,000

  We are assuming that 10,000 blocks are in the pools of both systems. We are also assuming that any block that is cached in the structure also resides in at least one of the buffer pools.

- Average block size of cached blocks = 4 KB

This implies that we have the following values for our calculations:

A 1000; Number of cached blocks

B 90,000; Number of uncached blocks that will be in the OSAM buffer pools at any time

C 2; Average number of 2 KB buffers required to hold a cached block

Then, we have:

```
Structure Size = 200(A + B) + (A x (2K x C)) + D
               = 200(1000 + 90,000) + (1000 x (2K x 2)) + 70K
               = 200(91,000) + (1000 x 4K) + 70K
               = 18,200,000 + 4,096,000 + 71,680
               = 22,367,680
               = 21,844K
               = 22,016K (rounded up to 256K increment)
```

The *dirratio* and *elemratio* subparameters of the CFOSAM parameter are then calculated as follows.

```
dirratio = A + B
         = 1000 + 90,000
         = 91,000

elemratio = A x C
          = 1000 x 2
          = 2000
```

Since these parameters must be calculated as one- to three-digit numbers, we could divide them both by 1000, yielding:

```
dirratio = 91

elemratio = 2
```

#### 15.1.4.4 DEDB VSO Structures

A user can choose to have either one or two cache structures for any shared DEDB area. Structures contain data from only one area. If two structures are used, they contain identical data. The second structure is used to provide increased availability. Since these structures do not support rebuild, the loss of the only structure for an area would require that the area be recovered using Image Copy, Change Accumulation, and IMS logs inputs. That is, the Database Recovery utility must be executed. Having two structures allows use of the area, including data-sharing, to continue when one structure fails or when connectivity to a structure is lost.

When two structures are used without the PRELOAD option, they must be the same size. If they are not, open of the area will fail.

When the PRELOAD option is used, the structure(s) must be large enough to hold all cached CIs in the area. If PRELOAD is not used, the structure(s) can be smaller. In this case, data is maintained in the structure(s) on a *least recently used* basis.

The first CI (CI0), CIs in the REORG UOW, and those used for SDEPs are not cached in the structure.

There is minimal benefit to having the two structures for a VSO area on the same CF. If the CF fails, both are lost. This has implications for the preference list (PREFLIST) in the CFRM policy. If there are only two CFs available in the Parallel Sysplex, an installation might not want to code both CFs in the preference list for both structures. If one CF were not available, both structures would be built on one CF. This would provide minimal availability advantages while suffering the overhead of maintaining two structures.

***Structure Size for Areas with PRELOAD Option:*** Structures for areas using PRELOAD must be large enough to hold all CIs in the direct portion. This does not include CI0 or the REORG UOW CIs.

Structures must hold the directory entries for each CI stored there. These are about 200 bytes each. The structure size for areas using PRELOAD can be calculated from the DBD AREA macro parameter values. The following shows the calculation:

```
AREA DD1=ddname,UOW(a,b),ROOT=(c,d),SIZE=e
```

Where the following values are relevant:

a # of CIs per UOW

c # of UOWs in direct portion

e CI size, rounded up to a multiple of 4 KB

The minimum structure size in bytes is:

```
SIZE = (1 + (a x c)) x (e + 200) + 70K
```

Assume that the following AREA statement is used:

```
AREA DD1=ABC001,UOW=(10,2),ROOT=(1000,100),SIZE=4096
```

The structure size would be calculated as follows:

```
SIZE = (1 (10 x 1000)) x (4096 + 200) + 70K
     = (10,001) x (4296) + 71,680
     = 43,035,976 bytes
```

Rounding up to a 256 KB boundary would result in a structure size of 42,240 KB.

***Structure Size for Areas without PRELOAD Option:*** Structures not using the PRELOAD option do not have to be large enough to hold all of the CIs in the direct portion. They must be large enough to hold information about all unique CIs that will either be in a local private buffer pool or cached in the structure concurrently. The size can be calculated as follows:

```
SIZE = n x (s + 200)
```

Where

n # of CIs in private buffer pools or cached

s CI size, rounded up to a multiple of 4 KB

For example, if there were two private buffer pools with 500 buffers each and we wanted an additional 1000 CIs cached at any time, we would need information about no more than 2000 CIs. Actually, we would need less if the same CI were in both buffer pools. Using 2000 CIs in our example and assuming a 4 KB CI size, the calculation would be as follows:

```
SIZE = 2000 x (200 + 4096) + 70K
     = 2000 x 4296 + 71,680
     = 8,663,680 bytes
```

Rounding up to a 256 KB boundary results in a structure size of 8,704 KB.

### 15.1.5 IRLM

IRLM uses a lock structure to provide locking capability among the IMSs sharing data.

The space in the structure is divided into a lock table, record data areas, and some control information. The control information uses about 28 KB of storage. The record data is used to hold modify locks. One does not directly tell IRLM the size of the lock table desired. IRLM infers this from other parameters in the IRLM procedure. IRLM attempts to use approximately one-half of the structure space for the lock table. The width of the lock table depends on the number of IRLMs, and the number of lock entries is required to be a power of 2. This can cause some unexpected results.

It is desirable to have a lock table that is large enough to keep false lock contention to a small number (less than 0.1 percent). There must be sufficient record space to hold the modified locks, or transactions will be abended (not a good thing). We will see that a structure with an adequately sized lock table will have plenty of room for modified locks in its record space.

IRLM is quite forgiving regarding the specification of the number of IRLMs. If more IRLMs connect to the structure than originally specified in the IRLM procedure, IRLM will rebuild the lock structure as required. This will effectively increase the width of a lock table entry, thus reducing the number of lock table entries.

The IRLM attempts to use half of the structure for the lock table. Since the lock table size must be a power of 2, this is not always possible. In these cases, the IRLM will choose a size that is as close as possible to half of the structure size while still being a power of 2. For example, if a structure is defined as 5 MB, the lock table will be 2 MB.

The number of entries in the lock table will depend on the size of each entry. These entries can be either 2, 4, or 8 bytes each. The size depends on two factors: the MAXUSRS value of the first IRLM to connect to the structure and the maximum number of IRLMs to ever be connected to the structure.

If the MAXUSRS value is 1 through 7, the entry size will be 2 bytes. If the MAXUSRS value is 8 through 23, the entry size will be 4 bytes. If the MAXUSRS value is 24 or greater, the entry size will be 8 bytes. These entry sizes can be adjusted when additional IRLMs connect to the structure. If MAXUSRS is 1 through 7, but the maximum number of IRLMs actually connected to the structure is 7 through 22, the entry size will be 4 bytes. If MAXUSRS is 1 through 23, but the maximum number of IRLMs connected to the structure is 24 or more, the entry size will be 8 bytes. The number of entries is easily calculated. It is the size of the lock table divided by the entry size. For a 2 MB table with 4 byte entries, the number of entries will be 512 KB.

The part of the structure not used by the lock table or control information is used for the record data area—that is, for modified locks. Each entry in this area requires approximately 140 bytes. If the record data area is 2,069,152, there would be space for approximately 14,779 modified locks.

### 15.1.5.1 The Simple Estimate

A simple approach to calculating the size of the IRLM lock structure is to begin with the maximum number of locks that you expect the IMSs to ever hold at one time. The structure size needed for an expected false contention rate of 0.1% is then calculated as follows.

```
  IRLM Structure Size = 2 x 1000 x Max. # Locks x Entry size
```

This size should be *rounded up to a power of 2*.

The entry size is 2 bytes for an IRLM MAXUSRS value of 1 to 7, 4 bytes for a MAXUSRS value of 8 to 23, and 8 bytes for MAXUSRS values of 24 or higher.

This calculation creates a structure such that half is used for the lock table and half is used for record data (modify locks). Since the lock table will have at least 1000 two-byte entries for each lock held, there will always be room for the modified locks in the record data storage. Modified locks are a subset of all locks, and they require only 140 bytes of storage.

### 15.1.5.2 The Simplest Estimate

The simplest estimate for the IRLM lock structure size is to just make it 32 MB. It is very rare for an installation to need a larger structure. A 32 MB structure will typically produce a false contention rate of approximately 0.1% when 8,000 locks are held by six or fewer IRLMs. An installation can create a 32 MB lock structure, monitor its usage, and adjust its size as required.

### 15.1.5.3 Calculating Structure Characteristics

No matter what structure size you use, you can use the following steps to calculate the number of entries in the lock table and in the record data area.

1. Take the structure size (SS) and divide by 2.

   SS / 2

2. Find the power of 2 that is closest to this—that is, closest to SS/2. This is the lock table size (TS)

3. Find the size of the entries (TE) in the lock table. For 1 through 6 IRLMs, this is 2. For 7 through 22, it is 4. For 23 or more, it is 8.

4. The number of lock table entries (#TE) is the lock table size (TS) divided by the size of the lock table entries (TE).

   #TE= TS / TE

5. The record data area size (RS) is the structure size (SS) minus the sum of the lock table size (TS) and the control information, 28 KB.

   RS = SS - TS - 28K

6. The number of record data area entries (RE) found by dividing the record data area size (RS) by the size of an entry, which is 140 bytes.

   NRE = RS / 140

The following is an example calculation:

Assume that we have a 5 MB structure with nine IRLMs.

1. SS / 2 = 5M / 2 = 2.5M

2. TS = 2M
3. TE = 4
4. #TE = TS / TE = 2M / 4 = 0.5M = 512K
5. RS = SS - TS - 28K = 5M - 2M - 28K = 3M - 28K = 3,044K
6. #RE = 3,044K / 140 = 22,264

## 15.1.6 Shared Queues

Shared queues uses list structures to hold IMS messages. Full-function queuing has a set of structures. Fast Path EMH has another set. These sets consist of a primary structure and an optional overflow structure. When a primary structure approaches its space limitation, queues can be moved to the overflow structure. Both kinds of structures have initial sizes and can be altered up to a maximum size. For this reason, shared queues is somewhat forgiving if the initial structure sizes are not accurately estimated.

List structures have many parts, including list headers, list entry controls, lock tables, adjunct areas, data entries, data elements, and event monitor controls. All of these require space in the structure. The text of messages are stored in the data elements. The other parts of the structure are used to control the messages. Making detailed estimations of the space required for each of these structure parts can be laborious and unproductive. Instead, we recommend making a simple estimation for the initial allocation of these structures.

### 15.1.6.1 Full-Function Message Queue Structures

The primary message queue structure should be large enough to hold the expected maximum number of messages. This size can be estimated in one of several ways. The use of IMS message queues before the implementation of shared queues can be used for this estimate. With traditional queuing, IMS stores messages in short and long message queue data sets. The sizes of these data sets give an upper limit to the space required for the messages. Since messages are stored in fixed length logical records, some space might not be used. Thus, the data sets might be considerably larger than the size of the messages might suggest.

A more refined estimate can be done by using the data that IMS maintains on the highest number of records used in the message queue data sets. These numbers can be found by examining log records. They are written in the statistics records which IMS writes to its log during system checkpoints. Fields STQDDLO and STQLMR in the x′4502′ log records contain the highest record numbers used for the short and long message queue data sets. Installations with the IMS/ESA Performance Analyzer product can use the IMS Internal Resource Usage report, which reports the *Highest SHMSG SHMSG block number ever used* and the *Highest LGMSG block number ever used*. Others can examine the x′4502′ log records by other means. Since shared queues stores all messages in its structures, the combined size of the messages stored in the small and large message queues should be used.

Once an estimate of the maximum total size of the the messages is made, the size of the structures can be determined. As a starting point, we recommend that the initial size (INITSIZE) of the the primary structure be at least twice the maximum total size of the messages. This provides an allowance for the parts of the structure other than the data elements. These parts include the list headers, list entry controls, and event monitor controls. The maximum size

(SIZE) of the structure can be considerably larger. As a starting point, a SIZE that is double INITSIZE could be used.

An overflow structure should be defined. It is only allocated if needed. That is, it is allocated when the primary structure has been altered to its maximum size and the threshold (OVFLWMAX) has been reached. Queues using 20 percent of the primary structure will be moved to the overflow. This implies that the overflow must be larger than 20 percent of the primary′s maximum size. We recommend that the size of the overflow structure should be at least 40 percent to 50 percent of the maximum primary structure size. For installations with a history of frequent message queue build ups, a large size should be used. This can be a size equal to the primary structure size.

After implementing shared queues, you should monitor the space used in your structures. This can be done by issuing a the following command:

```
  /CQQ STATISTICS STRUCTURE ALL
```

The response lists the number of data elements allocated (ELMALLOC) and the number of data elements in use (ELEMINUSE). This can be used to discover how much of the structures are actually being used. Experience will dictate whether the structure sizes should be reduced or increased.

#### 15.1.6.2 EMH Queue Structures

The primary EMH queue structure should be large enough to hold the expected maximum number of EMH messages on the queue at any time. This includes both input and output messages. The number of messages is limited by the number of terminals simultaneously submitting EMH messages. Additionally, messages selected for local processing are not placed in the structures. These considerations imply that EMH structures will typically be much smaller than those used for full-function messages.

As a starting point, installations should estimate the maximum number of EMH messages that will be in the system at any time. Each terminal can have a maximum of one input and one output message. Of course, it is unlikely that each terminal would actually have active EMH messages in the system at any time. The space required by the maximum number of messages can be estimated by multiplying the estimated average size times the estimated maximum number of messages. EMH messages have only one segment.

Next, you should estimate what fraction of your messages will actually be placed in the structure. These are the messages that are not processed locally. In typical installations, most messages will be processed locally. This information can be used to estimate an initial space requirement for EMH messages.

As with message queue structures, it is reasonable to begin by making the structure twice as large as the aggregate size of its messages. That is, leave half the space for the parts of the structure other than the data elements. Clearly, there are many variables that will determine the actual amount of space required by the structure. Most installations will not be able to easily determine all of these variables. So, we recommend a conservative approach of specifying a large size when in doubt.

The initial size (INITSIZE) should be set so that it easily handles your expected requirements. The maximum size (SIZE) should be set to handle a much larger number of messages. Of course, you can also specify an overflow structure. It is allocated when the primary structure has been altered to its maximum size and the threshold (OVFLWMAX) has been reached. Queues using 20 percent of the primary structure will be moved to the overflow. This implies that the overflow must be larger than 20 percent of the primary′s maximum size. We recommend that the size of the overflow structure should be at least 40 percent to 50 percent of the maximum primary structure size.

After implementing shared EMH queues, you should monitor the space used in your structures. This can be done by issuing the following command:

```
/CQQ STATISTICS STRUCTURE ALL
```

The response lists the number of data elements allocated (ELMALLOC) and the number of data elements in use (ELEMINUSE). This can be used to discover how much of the structures are actually being used. Once again, experience will dictate whether the structure sizes allocated should be decreased or increased.

## 15.1.7 Summary of Structure Characteristics

Table 5 summarizes the characteristics of the structures used by IMS in a Parallel Sysplex.

*Table 5. Structure Characteristics*

| Structure | Number | Structure Type | Recovery Method | Rebuild Support | Alter Support | Persistence | |
|---|---|---|---|---|---|---|---|
| | | | | | | Connection | Structure |
| **IRLM** | 1 | Lock | Rebuild | Yes | Yes[1] | Yes | Yes |
| **VSAM** | 1 | Directory-Only Cache | Rebuild | Yes | No | No | No |
| **OSAM** | 1 | Store-Through Cache[2] | Rebuild | Yes | No | No | No |
| **DEDB VSO** | 1 or 2 per Area | Store-In Cache | Dual Structures | No | No | Yes | No |
| **Shared Msg Q** | 1 | List | Rebuild | Yes | Yes | Yes | Yes |
| **Shared Msg Q Overflow** | 1 Optional | List | Rebuild | Yes | Yes | Yes | Yes |
| **Shared EMH Q** | 1 | List | Rebuild | Yes | Yes | Yes | Yes |
| **Shared EMH Q Overflow** | 1 Optional | List | Rebuild | Yes | Yes | Yes | Yes |

1. Alter cannot change the number of entries in the lock structure′s lock table.
2. If elemratio is zero, structure type is Directory-Only Cache.

## 15.2 Changing CF Structure Sizes

The size of structures might need to be changed. This is done by rebuilding the structures. Procedures should be established for changing structure sizes. There are various reasons that a structure′s size should be changed. For example, the addition of database buffers might require a larger buffer invalidation structure. Similarly, an increased workload may cause increased locking rates, which require a larger lock structure.

The steps required to change a structure′s size depend on the persistence attributes of a structure. These are explained below.

### 15.2.1 Connection and Structure Persistence

The attribute of persistence applies both to structures and to connections to them. These persistence attributes affect the disposition of a connection when a connector fails and the structure when there are no longer any connections to it.

#### 15.2.1.1 Connection Persistence

Connection persistence affects what happens when a connection to a structure ends abnormally. If the connection remains defined to the structure, it is a failed-persistent connection. A connection is persistent if its connection disposition is KEEP. It is not persistent if its connection disposition is DELETE.

Structures cannot be deleted when they have any defined connections to them. These can be active or failed-persistent.

#### 15.2.1.2 Structure Persistence

When a structure is allocated with a structure disposition of KEEP, it is persistent. When a structure is allocated with a structure disposition of DELETE, it is not persistent. Structures which are not persistent are automatically deleted by the system when there are no active or failed-persistent connections to them. Persistent structures are not automatically deleted in these circumstances. They remain allocated in the Coupling Facility. An operator must issue a command to delete them.

### 15.2.2 IMS Buffer Invalidation Structure Changes

The OSAM and VSAM buffer invalidation (cache) structures are not persistent structures and do not have persistent connections. This means that when all connections to them are removed, the structures are deleted by the system. The next time that they are built, they will be built with the specifications in the current CFRM policy.

This structure can be rebuilt while it is in use. When this is done, the size in the current CFRM policy will be used. Since rebuilding the structure while it is in use is disruptive to the system, this is not recommended as a normal procedure to change its size. Rebuilds of a VSAM structure cause all VSAM buffers in all IMS subsystems to be invalidated. Rebuilds of an OSAM structure cause all OSAM buffers in all IMS subsystems to be invalidated and cause all data cached in the structure to be discarded from the structure.

Before entering a command to rebuild a structure, it is advisable to display its current status. This will show if the structure is currently allocated, where it is allocated, and the IMS subsystems currently connected to it. The display command is:

```
DISPLAY XCF,STRUCTURE,STRNAME=structurename
```

A rebuild can be done by issuing the following command:

```
SETXCF START,REBUILD,STRNAME=structurename
```

### 15.2.3 DEDB VSO Cache Structure Changes

DEDB VSO cache structures are not persistent structures but have persistent connections. This means that a structure is automatically deleted from a Coupling Facility when all IMS subsystems using it have either terminated normally or processed a /DBR or /STOP AREA command. These actions terminate a connection normally. A subsystem failure or a CF link failure will result in a failed persistent connection. This will cause the structure to remain allocated.

Rebuilds of these structures are not supported. Their sizes cannot be changed while they have any connections, including failed persistent connections. To change a size, all connections must be terminated normally. When the area is opened again, the structure(s) will be built using the then current policy definition(s).

If duplicate structures are used without the PRELOAD option, they must be the same size. Opening an area will fail if they are not.

If duplicate structures are used with the PRELOAD option, they both must be large enough to hold the area.

### 15.2.4 IRLM Lock Structure Changes

IRLM lock structures are persistent structures with persistent connections. This means that they are not automatically deleted from a Coupling Facility when all IRLM connections to them are removed. To delete them, the operator must issue a SETXCF FORCE command.

An IRLM lock structure contains a lock table and a record list (sometimes called the modify lock list). The number of entries in the lock table can be recalculated when a lock table is rebuilt. This is true if either the size of the structure changes or the size (width) of the lock table entries change.

The number of entries in a lock table will affect the number of false contentions that occur. If there are too many false contentions, an installation should increase the number of entries. This is done by increasing the size of the structure.

This structure can be rebuilt while it is in use. When this is done, the size in the current CFRM policy will be used, and the number entries in the lock table will be recalculated.

Since rebuilding the structure while it is in use is disruptive to the system, this is not recommended as a normal procedure to change its size.

Before entering a command to rebuild a structure, it is advisable to display its current status. This will show if the structure is currently allocated, where it is allocated, and the IMS subsystems currently connected to it. The display command is:

```
DISPLAY XCF,STRUCTURE,STRNAME=structurename
```

A rebuild can be done by issuing the following command:

```
SETXCF START,REBUILD,STRNAME=structurename
```

An IRLM lock structure can be altered; however, there is a significant restriction in this support. Alter does not change the number of entries in the lock table. This means that altering the lock structure to a larger size will not change the number of false contentions. Alter only changes the amount of space used for the record list. A rebuild should be used to change the number of entries in the lock table.

Care should be exercised before deleting a lock structure. Do not delete one that contains locks protecting updates.

Before attempting to delete a structure, the operator should display its current status. This can be done with the following command:

```
DISPLAY XCF,STRUCTURE,STRNAME=structurename
```

If there are connections to the structure, they must be deleted first. The following command can be used:

```
SETXCF FORCE,CONNECTION,STRNAME=structurename,CONNAME=ALL
```

If there are no connections to the structure, it can be deleted with the following command:

```
SETXCF FORCE,STRUCTURE,STRNAME=structurename
```

### 15.2.5 Automatic Rebuilds

An increase in the number of IRLMs connected to a lock structure might cause it to be rebuilt automatically. This rebuild should be avoided since it is disruptive and could cause other problems.

The IRLM MAXUSRS parameter does *not* determine the maximum number of connections to the structure. Instead, it determines the size of the lock table entries when the structure is built. Values of 1 to 7 produce 2-byte entries. Values of 8 to 23 produce 4-byte entries. Values of 24 to 32 produce 8-byte entries. When an IRLM connects to a structure, it might cause the structure to be rebuilt with bigger lock table entries. If the structure has 2-byte entries, the rebuild is done when the seventh IRLM connects to the structure. This could be surprising. When MAXUSRS of 7 is specified, the seventh connector, not the eighth, causes the rebuild. The structure is rebuilt with 4-byte entries. Similarly, if the structure has 4-byte entries, it is rebuilt with 8-byte entries when the 23rd IRLM connects. When these rebuilds occur, the number of entries in the lock table is reduced. The space used for the lock table does not change. Since each entry in the lock table has doubled in size, the number of entries will be halved. This could cause increased false contentions.

Since rebuilding a structure is disruptive and since an increase in the number of IRLMs using a structure might increase the number of false contentions, always specify a MAXUSRS equal to or greater than the maximum number of IRLMs that will ever join the data-sharing group. If you will have 7 or 23 IRLMs, specify MAXUSRS of 8 or 24 to avoid the rebuild when the seventh or 23rd IRLM connects to the structure.

### 15.2.6 Shared Queues Structure Changes

Shared queues structures are persistent structures with persistent connections. This means that they are not automatically deleted from a Coupling Facility when all CQS connections to them are removed. To delete them, the operator must issue a SETXCF FORCE command. The messages in the structures can be recovered from the structure recovery data sets (SRDS). To delete the messages, these structure recovery data sets also must be deleted.

Shared queues uses two pairs of structures. One pair is for the IMS full-function message queues. The other pair is for Fast Path EMH messages. A pair consists of a primary structure and an overflow structure. Overflow structures are optional. The support for overflow processing, alter, and rebuild is the same for both pairs.

When the percentage of primary structure data elements exceeds a user specified threshold, overflow processing is invoked. If the CFRM policy contains an INITSIZE for the structure and the size has not yet reached the SIZE value, an alter of the structure is done. The size is increased to reduce the percent of data elements in use by 20. For example, if the threshold were 70 percent, the alter would increase the size of the structure and the number of data elements so that only 50 percent of the data elements would be in use. This process is repeated if the threshold is reached again. When the size of the structure reaches the CFRM policy SIZE specification, overflow processing builds the overflow structure and moves some messages to it. This assumes that the optional overflow structure has been requested. Movement of more messages will occur if the primary structure reaches the threshold again. This will continue until both structures are full. Eventually, puts to the structures might have to be rejected due to lack of space for them.

Overflow processing is based on the number of data elements in use. If there are plenty of data elements, but all list entries are in use, no more messages can be placed in the structure. Nevertheless, overflow processing will not be invoked.

### 15.2.7 Alter and Rebuild for Shared Queues Structures

Alter is used to change the size of a shared queues structure. Rebuild is used to move it or recover it.

Alter can be invoked for a shared queues structure either by overflow processing or by operator command. Alter will not change the ratio of data elements to list entries. Alter will only change the size of a structure. This size is limited by the SIZE parameter for the structure in the CFRM policy.

An alter can be done by issuing the following command.

```
SETXCF START,ALTER,STRNAME=structurename,SIZE=newsize
```

A rebuild of a shared queues structure will not change its size. Even if the INITSIZE or SIZE parameter in the CFRM policy is increased, rebuild will not change the size. Only alter can be used to change the size of these structures.

# Part 6. Operation Considerations

# Chapter 16. IMS Connections, Security and User Exits

This chapter describes the different types of IMS connections that exist, the need for security to protect your system, and a brief explanation about the different user exits that are available for your use.

## 16.1 IMS Connections

When migrating from a single IMS to a multiple IMS environment, provisions must be made for the subsystems, intelligent workstations and other connectors to the existing IMS. As part of migration planning, these connections must be identified for the existing environment so that provisions can be made for satisfying their requirements in the target environment.

Because there is no longer a single IMS communicating with its connectors, some changes might be required not only to the target IMS but also to the connectors. For example, if the existing IMS were connected a DB2 subsystem, each clone of it will have to be connected to a DB2 subsystem. For APPC conversations, SIDE_INFO and TP_PROFILE data sets might require updating. Each connection should be evaluated for its requirements in the target environment.

Table 6 identifies the more common types of connections found in IMS environments. In this table, the type of connector and the type of connection are identified.

*Table 6 (Page 1 of 2). Summary of IMS Connectors*

| CONNECTOR TYPE | CONNECTION TYPE | DESCRIPTION |
|---|---|---|
| IMS | MSC | Existing IMS is connected through MSC to another IMS. |
| IMS | ISC | Existing IMS is connected through ISC to another IMS. |
| CICS | ISC | Existing IMS is connected through ISC to a CICS region. |
| CICS | DBCTL | Existing IMS is providing DBCTL services to a CICS region. |
| DB2 | ESS | Existing IMS is connected to a DB2 subsystem. |
| LU62 | APPC-IMP-IN | Existing IMS receives implicit APPC transactions from APPC applications. |
| LU62 | APPC-IMP-OUT | Existing IMS runs applications which issue CHNG/ISRT calls to APPC destinations. |
| LU62 | APPC-EXP-IN | Existing IMS runs applications which ACCEPT explicit conversations from APPC applications. |
| LU62 | APPC-EXP-OUT | Existing IMS runs applications which ALLOCATE explicit conversations with APPC applications. |
| MQSeries | ESS and BMP | Existing IMS receives MQSeries input using MQ Calls and the Trigger Monitor BMP. |
| MQSeries | OTMA | Existing IMS receives MQSeries input through MQSeries IMS Bridge using OTMA. |

*Table 6 (Page 2 of 2). Summary of IMS Connectors*

| CONNECTOR TYPE | CONNECTION TYPE | DESCRIPTION |
|---|---|---|
| Web Server | OTMA | Existing IMS receives transactions from the Web through the IMS TCP/IP OTMA Connection (ITOC). |
| Web Server | APPC | Existing IMS receives transactions from the Web through an APPC connection. |
| JAVA | OTMA | Existing IMS receives transactions from a JAVA application through the IMS TCP/IP OTMA Connection (ITOC). |
| TCP/IP Sockets | BMP | Existing IMS uses a BMP to provide a sockets interface. |
| TCP/IP | OTMA | Existing IMS uses ITOC for sockets connections. |
| DCE RPC | ISC | Existing IMS receives RPC requests through DCE Application Server′s ISC interface. |
| DCE RPC | APPC | Existing IMS receives RPC requests through DCE Application Server′s APPC interface. |
| DCE RPC | OTMA | Existing IMS receives RPC requests through DCE Application Server′s OTMA interface. |
| ODBA | DRA | Provides MVS batch application program access to IMS TM or DBCTL databases. |

## 16.2 Security

It is the responsibility of the installation to provide the security environment for the couple data sets. RACF, or an equivalent security product, can be used to protect access to system and database data sets.

### 16.2.1 RACF Security

For protected resources, users must be authorized through RACF. Cloned systems should have equivalent access to IMS resources; therefore, they should have the same RACF RCLASS as defined on the system definition SECURITY macro.

Coupling Facility structures may be protected using RACF or an equivalent security product. Structures are protected by using a RACF resource profile of IXLSTR.structure-name in the FACILITY class. CQS structures can also be protected by resource profile of CQSSTR.structure-name. The differences in the use of these two profiles of CQS structures is explained below.

The following is an example of the steps to protect an OSAM structure with name IMSP_IMSOSAM. They allow only users with USERID of IMSP to connect to the structure and manipulate it. Users would include IMS subsystems and operators who issue SETXCF commands for the structure.

1. RDEFINE FACILITY IXLSTR.IMSP_IMSOSAM UACC(NONE)
2. PERMIT IXLSTR.IMSP_IMSOSAM CLASS(FACILITY) ID(IMSP) ACCESS(ALTER)
3. SETROPTS CLASSACT(FACILITY)

Similarly, the following steps could be used to protect an IRLM lock structure with name IMSP_IMSIRLM. They allow only users with USERID of IRLMP to connect to the structure and manipulate it. Users would include IRLMs and operators who issue SETXCF commands for the structure.

1. RDEFINE FACILITY IXLSTR.IMSP_IMSIRLM UACC(NONE)
2. PERMIT IXLSTR.IMSP_IMSIRLM CLASS(FACILITY) ID(IRLMP) ACCESS(ALTER)
3. SETROPTS CLASSACT(FACILITY)

The following steps could be used to protect a CQS structure with name IMSP_IMSMSGP. They allow only users with USERID of IRLMP to connect to the structure and manipulate it. Users would include IMSs, CQSs, and operators who issue SETXCF commands for the structure.

1. RDEFINE FACILITY IXLSTR.IMSP_IMSMSGP UACC(NONE)
2. PERMIT IXLSTR.IMSP_IMSMSGP CLASS(FACILITY) ID(IMSP) ACCESS(ALTER)
3. SETROPTS CLASSACT(FACILITY)

If a user does not have authority to access a structure, the following message is issued.

```
 IXL012I IXLCONN REQUEST FAILED, RETCODE: 00000008, RSNCODE: 0201084C
```

If RACF profiles are not defined, the default allows any authorized user or program (supervisor state and PKM allowing key 0-7) to issue Coupling Facility macros for the structure. For more information about RACF, see *OS/390 Security Server (RACF) Security Administration Guide*, SC23-3726.

CQS structures also can be protected by using a RACF resource profile of CQSSTR.structure-name. This only restricts CQS clients. When this profile is used, CQS clients without sufficient authority are denied access to the structure. On the other hand, other users of XES services are not denied by this profile. Since this profile is only effective for CQS clients and the IXLSTR profile is effective for all users, we recommend the use of the IXLSTR profile.

## 16.3 IMS SMU Security

IMS security is implemented with the IMS Security Maintenance Utility (SMU). Since cloned systems should have equivalent access to IMS resources, SMU input for them should be equivalent.

## 16.4 Database Data Set Dispositions

Since multiple systems have database data sets open concurrently, DISP=SHR is usually required for them. This rule is enforced for VSAM data sets since DISP=SHR is also required for CBUF processing. This is explained in 11.4, “Database Data Set Sharability” on page 112. The rule might not be explicitly enforced for OSAM. It is enforced in two situations:

1. Multiple allocations on one MVS system.

2. Multiple allocations on different MVS systems with GRS using a SYSTEM inclusion list containing SYSDSN. The default inclusion list supplied with GRS contains SYSDSN.

In these cases, allocation will fail if DISP=SHR is not specified by all users.

## 16.5 User Exits

Most user exits will not require any modification to function in an IMS Parallel Sysplex environment; however, any exit which is sensitive to the IMS environment, or to the network, should be reviewed to ensure that they will continue to function properly.

The following sections list considerations for the IMS exit routines.

### 16.5.1 IMS System Exits

The following is a list of the IMS system exit routines. Installations should examine each of the routines that they use to determine if they are appropriate for each of the IMS systems in their IMSplex.

Typically, the same routines would be used for each IMS system. For those routines that produce output files, such as the Log Archive Exit Routine, it might be necessary to combine output files from multiple IMS systems.

- Type 2 Automated Operator Exit Routine (DFSAOE00)
- Buffer Size Specification Facility (DSPBUFFS)
- Command Authorization Exit Routine (DFSCCMD0)
- Dependent Region Preinitialization Routines
- Dump Override Table (DFSFDOT0)
- IMS Command Language Modification Facility (DFSCKWD0)
- Large SYSGEN Sort/Split Input Exit Routine (DFSSS050)
- Large SYSGEN Sort/Split Output Exit Routine (DFSSS060)
- Log Archive Exit Routine
- Log Filter Exit Routine (DFSFTFX0)
- Logger Exit Routine (DFSFLGX0)
- Non-Discardable Messages Exit Routine (DFSNDMX0)
- Partner Product Exit Routine (DFSPPUE0)
- RECON I/O Exit Routine (DSPCEXT0)
- Resource Access Security Exit Routine (DFSISIS0)
- System Definition Preprocessor Exit Routine (Input Phase) (DFSPRE60)
- System Definition Preprocessor Exit Routine (Name Check Complete) (DFSPRE70)
- User Message Table (DFSCMTU0)

## 16.5.2 IMS Database Manager Exits

The following is a list of the IMS database exit routines. Installations should examine each of the routines that they use to determine if they are appropriate for each of the IMS systems in their IMSplex.

For shared databases, each system must use the same exit routines with two exceptions. First, the Data Capture Exit Routine might be different on different systems. For example, it might write changed data to different places. Second, the Sequential Buffering Initialization Exit might take different actions on different systems.

- Data Capture Exit Routine
- Data Entry Database Randomizing Routine
- Data Entry Database Resource Name Hash Routine (DBFLHSH0)
- Data Entry Database Sequential Dependent Scan Utility Exit Routine (DBFUMSE1)
- HDAM Randomizing Routines
- Secondary Index Database Maintenance Exit Routine
- Segment Edit/Compression Exit Routine
- Sequential Buffering Initialization Exit Routine (DFSSBUX0)

## 16.5.3 IMS Transaction Manager Exits

The following is a list of the IMS TM exit routines. Installations should examine each of the routines they use to determine if they are appropriate for each of the IMS TM systems in their IMSplex.

Typically, the same routines would be used for each IMS system, but there can be exceptions. For example, the Automated Operator Exit Routine might have different requirements in different systems. The Front-End Switch Exit Routine is likely to make different routing decisions in different IMS systems. The Sign-On Exit Routine might need to differ in each system, especially if it is used to handle the situations described in 8.4, “ETO Considerations with Shared Queues” on page 59. The TCO Exit routine is likely to require different scripts in different IMS systems.

- Type 1 Automated Operator Exit Routine (DFSAOUE0)
- Build Security Environment Exit Routine (DFSBSEX0)
- Conversational Abnormal Termination Exit Routine (DFSCONE0)
- Fast Path Input Edit/Routing Exit Routine (DBFHAGU0)
- Front-End Switch Exit Routine (DFSFEBJ0)
- Global Physical Terminal (Input) Edit Routine (DFSGPIX0)
- Greeting Messages Exit Routine (DFSGMSG0)
- Initialization Exit Routine (DFSINTX0)
- Input Message Field Edit Routine (DFSME000)
- Input Message Routing Exit Routine (DFSNPRT0)
- Input Message Segment Edit Routine (DFSME127)
- Logoff Exit Routine (DFSLGFX0)

- Logon Exit Routine (DFSLGNX0)
- LU 6.2 Destination Exit Routine (DFSLULU0)
- LU 6.2 Edit Exit Routine (DFSLUEE0)
- Message Control/Error Exit Routine (DFSCMUX0)
- Message Switching (Input) Edit Routine (DFSCNTE0)
- MSC Link Receive Routing Exit Routines (DFSCMLR0 and DFSCMLR1)
- MSC Program Routing Exit Routine (DFSCMPR0)
- OTMA Destination Resolution Exit Routine (DFSYDRU0)
- OTMA Input/Output Edit Exit Routine (DFSYIOE0)
- OTMA Prerouting Exit Routine (DFSYPRX0)
- Output Creation Exit Routine (DFSINSX0)
- Physical Terminal (Input) Edit Routine (DFSPIXT0)
- Physical Terminal (Output) Edit Routine (DFSCTTO0)
- Queue Space Notification Exit Routine (DFSQSPC0)
- Security Reverification Exit Routine (DFSCTSE0)
- Shared Printer Exit Routine (DFSSIML0)
- Signoff Exit Routine (DFSSGFX0)
- Sign-On Exit Routine (DFSSGNX0)
- Sign On/Off Security Exit Routine (DFSCSGN0)
- Terminal Routing Exit Routine (DFSCMTR0)
- Time-Controlled Operations (TCO) Exit Routine (DFSTXIT0)
- Transaction Authorization Exit Routine (DFSCTRN0)
- Transaction Code (Input) Edit Routine (DFSCSMB0)
- 2972/2980 Input Edit Routine (DFS29800)
- 4701 Transaction Input Edit Routine (DFS36010)
- 7770-3 Input Edit Routine (DFSI7770)
- 7770-3 Output Edit Routine (DFSO7770)
- 7770-3 Sign-on Exit Routine (DFSS7770)

### 16.5.4 Common Queue Server Exit Routines

The following is a list of the CQS exit routines. IMS TM supplies these exit routines, but a user could modify them. Typically, installations use the supplied exit routines in all of the systems in a shared queues group.

- Initialization-Termination Exit Routine
- Client Connection Exit Routine
- Queue Overflow Exit Routine
- Structure Statistics Exit Routine
- Structure Event Exit Routine
- Client CQS Event Exit Routine

- Client Structure Event Exit Routine
- Client Structure Inform Exit Routine

# Chapter 17. IMS and User JCL

All JCL used actively within or in support of the IMS environment must be reviewed for necessary changes. These changes are primarily the result of data set name (DSN) changes required due to the addition of multiple IMS subsystems with similar data sets. In some cases, the execution parameters specified in the JCL must be changed to reflect the data-sharing environment. The following are examples of some of the JCL that must be reviewed and possibly updated.

## 17.1 IMS Procedures

The IMS system definition process generates several PROCs and places them in IMS.PROCLIB. They are then (usually) tailored to the specific IMS environment and either moved to SYS1.PROCLIB or placed in a library which is concatenated with SYS1.PROCLIB. This library can be IMS.PROCLIB itself or some other library. For purposes of this discussion, we will assume that these PROCs are all in IMS.PROCLIB. See 5.7, "JCL Libraries" on page 34 for a discussion of libraries.

### 17.1.1 Started Procedures

These are procedures which are initiated using the `MVS START` command. Although it is possible to specify parameters when issuing this command, many of the parameters for the PROC must be included in the PROC itself, or available from a parameter library. For example, the MVS command `START IMSA` would start an IMS control region using the IMSA PROC found in IMS.PROCLIB. `START IMSA,PARM1='AUTO=Y'` would override the AUTO parameter in the IMSA PROC. The IMS control region would then start the DLI and DBRC address spaces using PROC names specified by IMSA's `DLINM` and `DBRCNM` parameters.

**IMS** This procedure is used to start the IMS TM control region. You will need one version of this PROC for each IMS control region you wish to run. For example, if your IMS systems have subsystem names of IMSA and IMSB, you probably want two PROCs named IMSA and IMSB.

Except for RGSUF, all parameters can be overridden by member DFSPBxxx in IMS.PROCLIB. RGSUF identifies the specific DFSPBxxx member to use. In addition, the member DFSDCxxx (new with IMS V6) can be used to override the Primary and Secondary Master terminal definitions set by STAGE1 input as well as to change the system default truncated data option (also new with IMS V6).

Several DD statements in the IMS procedure must be changed to conform to new naming conventions and to use new data set names.

**DBC** This procedure is used to start the IMS database control region (DBCTL). You will need one version of this PROC for each control region you wish to run. For example, if your DBCTL systems have subsystem names of DBCA and DBCB, you probably want two PROCs named DBCA and DBCB.

Except for RGSUF, all parameters can be overridden by member DFSPBxxx in IMS.PROCLIB. RGSUF identifies the specific DFSPBxxx member to use.

Several DD statements in the DBC procedure must be changed to conform to new naming conventions and to use new data set names.

**CQS** CQS is the default name for the PROC to start the CQS address space, which can be overridden in the DFSSQxx PROCLIB member. You will need one version of this PROC for each IMS control region that is to join a shared queues group.

A sample CQS PROC is not documented in the IMS manuals. An example is shown below:

```
//CQS       PROC   RGN=3000K,SOUT=A,
//                 BPECFG=BPECONFG,
//                 CQSINIT=,
//                 PARM1=
//CQSPROC   EXEC   PGM=CQSINIT0,REGION=&RGN,
//                 PARM=′BPECFG=&BPECFG,
//                 CQSINIT=&CQSINIT,&PARM1′
//STEPLIB   DD     DSN=IMSV6.RESLIB,DISP=SHR
//PROCLIB   DD     DSN=IMSV6.PROCLIB,DISP=SHR
//SYSPRINT  DD     SYSOUT=&SOUT
//SYSUDUMP  DD     SYSOUT=&SOUT
```

The PROCLIB member specified by the BPECFG= parameter in the CQS execution PROC defines configuration parameters to BPE. BPE is the mnemonic for Base Primitive Environment upon which CQS is built. IMS provides a default BPECFG member which can be used. One would likely change it if requested by IMS programming services.

The CQSINIT= specification points to three CQS-related PROCLIB members, CQSIPxxx, CQSSLxxx, and CQSSGxxx. CQSIPxxx includes parameters that are related to the initialization of a CQS address space. CQSSLxxx defines local CQS parameters that are related to one or more Coupling Facility structures. Each CQS should point to a different CQSSLxxx member. CQSSGxxx defines global CQS parameters that are related to one or more Coupling Facility structures. CQSSGxxx parameters are shared by all CQS address spaces that share a structure.

Documentation related to creating these PROCLIB members is in the *IMS/ESA Common Queue Server Guide and Reference*, SC26-9517.

**DLISAS** DLISAS is the default name for the PROC to start the DLI separate address space for both the IMS TM and DBCTL systems. The name of this procedure can be overridden by specifying DLINM in one of several places including the IMSCTRL MACRO, DFSPBxxx, and the IMS (or DBC) procedure itself. Because multiple IMSs can be running on one MVS (with one SYS1.PROCLIB concatenation), each IMS should have its own unique DLISAS PROC (for example, DLSA). This allows any IMS to run on any MVS and use the same PROCLIB.

When the control region issues the MVS START command for DLISAS, it will include its own IMSID (for example, IMSA). Therefore, it is not necessary to include the IMSID in the DLISAS PROC.

```
START DLSA,PARM=(DLS,IMSA)
```

Several DLISAS PROC DD statements must also be changed to conform to new naming conventions, and to new data set names. For example, the //STEPLIB (RESLIB), //PROCLIB, and //ACBLIBx DD statements will probably need to have their data set names updated.

**DBRC** DBRC is the default name for the PROC to start the DBRC address space. The name of this PROC can be overridden by specifying DBRCNM in several places, including the IMSCTRL MACRO, DFSPBxxx, and the IMS (or DBC) PROC itself. Because multiple IMSs can be running on one MVS (with one SYS1.PROCLIB concatenation), each IMS should have its own unique DBRC PROC (for example, DRCA). This allows any IMS to run on any MVS and use the same PROCLIB.

When the control region issues the MVS START command for DBRC, it will include its own IMSID (for example, IMSA). Therefore, it is not necessary to include the IMSID in the DBRC PROC.

```
START DBRA,PARM=(DRC,IMSA)
```

Several DBRC PROC DD statements must also be changed to conform to new naming conventions, and to new data set names. For example, the //STEPLIB (RESLIB), //PROCLIB, and //JCLPDS DD statements will probably need to have their data set names updated.

**IMSRDR** IMSRDR is the default name for the PROC that is invoked by IMS when the MTO issues the /START REGION command:

```
/START REGION member
```

which results in an MVS START command being issued

```
S IMSRDR,MBR=member
```

The name of this PROC can be overridden by specifying PRDR in one of several places, including the IMSCTRL MACRO, DFSPBxxx, and the IMS (or DBC) PROC itself.

The member name of the JOB being started must be in the library defined in the IEFRDER DD statement (for example, IMS.JOBS).

If no member name is specified, a default member name is used. This default name is specified in the IMSRDR PROC, and if not changed from what is generated by IMS system definition, is IMSMSG. IMSMSG, or the member name specified in the command, is a JOB found in IMS.JOBS. This job executes procedure DFSMPR to start an IMS message processing region. Because multiple IMSs can be running on one MVS (with one SYS1.PROCLIB concatenation), each IMS should have its own unique IMSRDR PROC (for example, RDRA), specifying its own unique default (for example, IMSMSGA). This allows any IMS to run on any MVS and use the same PROCLIB.

**DXRJPROC** Although this is the same procedure name for IRLM 1.5 and IRLM 2.1, the PROCs are somewhat different. Each IRLM must have a unique IRLMID (any value from 0 to 255) but have the same GROUP value. Since we recommend that there be only one IMS IRLM per MVS, the IRLM names should be the same for all. This will facilitate the restarting of one IMS on any MVS without having to change the IRLMNM.

### 17.1.2 Executed Procedures

These are procedures which are executed as jobs (meaning, not as started procedures). There can be any number of these PROCs which are in use with a given IMS system. The following are some of those that are created by IMS system definition and placed in IMS.PROCLIB.

**DFSMPR** This is the procedure executed by the IMSMSG jobs to start IMS TM message processing regions (MSG). There should be no reason to modify this PROC as all of the parameter overrides will be in the executed jobs member of IMS.JOBS (see 5.7, "JCL Libraries" on page 34).

**IMSFP** This is the procedure executed by the job which starts a Fast Path region (IFP). As with DFSMPR, there should be no reason to modify this PROC as all of the parameter overrides will be contained in the JOB JCL. Note that if IFP regions are to be started with /START REGION <member> commands, one or more Fast Path members (similar to IMSMSG) must be in IMS.JOBS. These jobs would then execute this procedure.

**IMSBATCH** This procedure is invoked for BMPs. Normally, the parameters in this procedure are set by the job which executes it. If this is the case, there are no parameters which must be changed.

**FPUTIL** Like DFSMPR, IMSFP, and IMSBATCH, this PROC is executed by a job which can provide all of the parameter and JCL overrides necessary. Therefore, it is not necessary to modify this PROC or to provide unique versions for each IMS.

**DFSWTnnn** This procedure is used by IMS TM to print spool SYSOUT data sets. Each procedure includes the DD statements for the spool SYSOUT data sets used for the *nnn* spool SYSOUT line.

## 17.2 IMS Jobs

When the /START REGION command is issued, the IMSRDR PROC, using the MVS internal reader (INTRDR), starts a JOB from a library specified in the IMSRDR JCL (such as IMS.JOBS) with the member name specified in the command or the default specified in the IMSRDR PROC. For example,

```
/START REGION MSGA01
```

results in a JOB in IMS.JOBS with a member name of MSGA01 being executed.

**IMSMSG** This is the default member name for the /START REGION command. It is specified in the IMSRDR PROC (or whatever name is specified with PRDR) and can be changed in that PROC. This is the job that is started when the /START REGION command is entered without a member name.

```
     /START REGION

  starts the IMSRDR PROC without overriding the MBR parameter

     //IMSRDR  PROC  MBR=IMSMSG,...
```

This JOB then executes a procedure for the type of region being started (such as a message processing region).

```
    //IMSMSG   JOB   ....
    //STEP01   EXEC  DFSMPR,IMSID=IMSA,.....
```

You will need a JOB for each member specified in the /START REGION <member> command. These should be the same JOBS for each target IMS as you have in your existing IMS. For example, you will want JOBS for several different MSG regions (with different parameters), and possibly some IFP and BMP regions so they can be started by the /START REGION command. With the use of either the LOCAL command parameter or the function introduced by APAR PQ21039, it is not necessary to have a different version of each JOB for each IMS system. Instead, either a /START REGION xxx LOCAL command can be used or the IMSGROUP (see 17.6.1, “Using the Function Delivered by APAR PQ21039” on page 173) parameter can be used by the control region.

**Fast Path** There is no default member for starting Fast Path (IFP) regions. IFP regions can be started either through normal JES scheduling, or by using the /START REGION <member> command. If the /START REGION <member> method is to be used, then the same considerations apply for this (these) jobs as for IMSMSG, except that they would execute an IMSFP procedure. The LOCAL parameter can be used with /START REGION command. The IMSGROUP parameter for the control region eliminates the need to specify different IMSID parameters for IFP regions using different control regions.

**BMPs** BMPs are usually started by submitting them through JES, but they can be started by using the /START REGION <member> command, where <member> is the member name of the BMP job in IMS.JOBS. Considerations for the /START command and the use of IMSGROUP are the same for BMPs as they are for MPP and IFP regions.

**IMSWTnnn** This JOB executes IMS TM procedure DFSWTnnn. There are no IMS parameters in this job. It merely specifies the corresponding DFSWTnnn procedure.

The installation does not have a choice about the name of the member in the IMS.JOBS. It must be of the form IMSWTnnn, where the *nnn* is created in the system definition process. If the system definition TERMINAL macro specifies FEAT=AUTOSCH, IMS will automatically issue a /STA REGION IMSWTnnn command when a spool SYSOUT data set is full. This requires that IMSWTnnn be in the IMS.JOBS data set. The IMSWTnnn job uses the DFSWTnnn procedure. This requires that DFSWTnnn be in the concatenation of SYS1.PROCLIB. The DD statements specifying the spool SYSOUT data sets are specified in the DFSWTnnn procedures.

If IMS TM Spooled SYSOUT is used, the DFSWTnnn procedures will have to differ for each IMS clone. This means that different systems will have to use procedure libraries.

## 17.3 DBRC Skeletal JCL

DBRC skeletal JCL, used by the GENJCL commands, are found in the DBRC JCLPDS data set. These include both IBM-supplied JCL members, as well as any user JCL (used in GENJCL.USER) that might have been created. The skeletal JCL members should be reviewed for any dependencies on an IMS system. Using the %SSID parameter to identify the IMS system is usually sufficient to allow the same member to be used for multiple IMS systems.

The following member types should be examined. For all members, consideration should be given to the MVS system on which they should be run and the initiator classes available on that MVS.

- JOB (JOBJCL)

  This member contains the JOB statement used by GENJCL commands. It can be modified to include accounting information or to add statements, such as a JOBLIB or a JES control statement.

- ARCHIVE (ARCHJCL)

  The IBM supplied member includes the %SSID parameter to generate JCL for each IMS system. The archive jobs for an IMS system can be run on any MVS system in the Sysplex.

- CHANGE ACCUMULATION (CAJCL)

  Change Accumulation is required for the recovery of shared databases. It can be run on any MVS system in the Sysplex.

- IMAGE COPY (ICJCL)

  Many installations do not use GENJCL to generate their Image Copy JCL. GENJCL.IC is not required for generating Image Copy jobs unless the database data set has REUSE specified for it in the RECONs.

  Image Copy can be run on any MVS system in the Sysplex.

- RECOVERY (RECOVJCL)

  Database Recovery can be run on any MVS system in the Sysplex.

- RECEIVE (ICRCVJCL)

  RSR image copy receive processing occurs at the remote site, not at the operational site. This is independent of data-sharing.

- ONLINE IMAGE COPY (OLICJCL)

  Online Image Copy (OLIC) can be run on any IMS system; however, the other IMS systems cannot have update intent while OLIC is being run for a database. Since OLIC is run as a BMP, it must specify or default to the IMSID of the IMS system running on the MVS where OLIC is executed. GENJCL.OLIC is not required for generating OLIC JCL unless the database data set has REUSE specified for it in the RECONs.

- LOG CLOSE (LOGCLJCL)

The Log Recovery utility can be run on any MVS system in the Sysplex. The IMS supplied member includes the %SSID parameter to identify the IMS system whose log is being closed.

- USER

  This is installation dependent. Members used by GENJCL.USER should be reviewed for any IMS system dependencies.

## 17.4 Other Backup and Recovery JCL

If DBRC GENJCL is not used to generate backup and recovery JCL, this JCL must be identified and reviewed. In general, database backups and recoveries can be run on any MVS system; however, an installation might want to direct these jobs to a specific MVS system for performance or operational reasons. Online Image Copy must run on the MVS system where its IMS control region is executing.

## 17.5 Other Application JCL

It is likely that many application processes will contain steps that do not access IMS at all, but do process data sets that will be used as input to IMS steps or that is output from IMS steps. For example, GSAM input and output files are often processed as sequential files by non-IMS steps.

## 17.6 BMP JCL

The IMSBATCH procedure contains much of the IMS-related JCL used by BMPs, but most BMP job steps also contain user JCL which must be reviewed. This section describes the JCL requirements other than user data set DD statements.

The IMSID parameter of the BMP′s JCL must identify the IMS system to which it will be connected. That is, it specifies the IMS control region to be used by the BMP. This will be different for each IMS system in an IMSplex.

The default value for the IMSID is obtained from module DFSVC000 in RESLIB. The value in DFSVC000 is set by the IMSID parameter of the IMSCTRL macro in IMS′s system definition. When a new system definition is done, it will create a new DFSVC000 module with the latest IMSID specification.

### 17.6.1 Using the Function Delivered by APAR PQ21039

APAR PQ21039 for IMS 6.1 simplifies the specification of the IMSID for users whose BMPs might execute on multiple control regions in the IMSplex. With this enhancement, no changes to BMP JCL are required, even when different control regions are used. The meaning of the IMSID parameter has been expanded by a change in the control region parameters. IMSID now indicates either a control region ID or an IMS control region group name. IMSGROUP is a new control region parameter. Each control region in an IMSplex can specify the same value. When cloning, it is expected that this name will be the name that was formerly the IMSID of the control region.

Table 7 on page 174 and Table 8 on page 174 illustrate a use of the IMSGROUP parameter.

*Table 7. IMSID Usage Before Cloning*

| Control Region | Dependent Regions |
|---|---|
| IMSID=IMSP | IMSID=IMSP |

*Table 8. IMSID Usage After Cloning*

| MVS1 Control Region | MVS2 Control Region | Dependent Regions |
|---|---|---|
| IMSGROUP=IMSP<br>IMSID=IP11 | IMSGROUP=IMSP<br>IMSID=IP12 | IMSID=IMSP |

It is important to note that dependent region JCL does not have to change. Only control region JCL is changed to exploit the function supplied by IMSGROUP.

## 17.6.2 The IMSGROUP Function

When IMSGROUP is specified for an IMS control region, IMS builds an MVS name/token pair whose name is based on the IMSGROUP specification and the IMS version and release. It stores its IMSID in the token. Dependent regions can examine the token to discover the control region′s IMSID.

Dependent regions (MPP, BMP, and IFP) use the following logic to determine the IMSID to use.

1. The dependent region attempts to connect to a control region whose subsystem name matches the IMSID specification of the dependent region. If there is no such control region on this MVS, the search continues.
2. If ALTID is specified for the dependent region, it attempts to connect to a control region whose subsystem name matches the ALTID specification of the dependent region. If there is no such control region on this MVS, the search continues.
3. The dependent region looks for an MVS name/token pair whose name matches its IMSID and IMS version and release. If its finds such an MVS name/token pair, it finds the IMSID stored there and connects to it. If there is no such MVS name/token, the search continues.
4. If ALTID is specified, the dependent region looks for an MVS name/token whose name matches its ALTID and IMS version and release. If it finds such an MVS name/token pair, it finds the IMSID stored there and connects to it. If there is no such name/token pair, message DFS690A is issued. The text of the message is:

```
DFS690A job.step.proc - CTL PGM NOT ACTIVE, REPLY ′WAIT′, ′CANCEL ′
OR ′ALT-ID′
```

## 17.6.3 Handling IMSID Without APAR PQ21039 Function

The use of the function delivered by APAR PQ21039 is highly recommended. For installations using IMS/ESA 5.1, this function is not available. The following considerations apply to such users.

Different installations might handle the IMSID specification for BMPs differently. The following cases can exist:

- A BMP is run on only one IMS system.

Some installations will want to run a BMP on only one system. For example, they might want all batch work to run on a large processor. If they have processors with different capacities, they might want the fastest processor to handle all of their batch work. These installations would route all BMP jobs to the MVS system on the fastest processor. They have two choices for specifying the IMSID. First, they could use the default specified in a RESLIB. All BMPs would use the same RESLIB. This RESLIB would have the appropriate IMSID. Second, they could specify the IMSID in the execution parameters. All BMPs would use the same IMSID.

- A BMP is run on only two IMS systems

  The BMP parameters include ALTID. ALTID specifies an alternate IMSID. A BMP will attempt to use the IMS system specified on its IMSID parameter. If an IMS system with this IMSID is not present on the MVS system, the BMP will attempt to use the IMS system specified by the ALTID. This allows a BMP to specify a first and second choice for IMS systems. There is no default for the ALTID. It must be specified in the BMP′s JCL. In this case, the BMPs would specify both the IMSID and the ALTID execution parameters.

- A BMP can be run on more than two IMS systems

  If there are more than two IMS systems on which a BMP can be run, its IMSID might have to be modified with each execution. The modification might be made in four ways. First, the IMSID JCL parameter can be changed. Second, the library containing the default IMSID in its DFSVC000 member can be changed. Third, automated operations can be used to change the IMSID dynamically. Fourth, the LOCAL parameter on the /START REGION command can be used.

  – The first method, changing the JCL, is obvious but labor intensive.

  – The second method, changing the library, can be done by one of the following techniques:

    - Changing the JCL to point to another library in the STEPLIB concatenation. This is as labor intensive as changing the IMSID execution parameter.

    - Using LNKLST to find the IMS RESLIB which contains the DFSVC000 module. In this scheme, each MVS system has a different LNKLST. The LNKLST is used to locate the IMS RESLIB for the IMS systems and BMPs. That is, BMPs and IMS systems do not use JOBLIB or STEPLIB. The BMPs do not specify the IMSID in there execution parameters. The BMPs use the default IMSID specified in the DFSVC000 module of their RESLIB. Each MVS system must have a LNKLST pointing to an IMS RESLIB with a default IMSID that matches the IMS system executed on the MVS. If an IMS system is moved to another MVS system, BMPs will not use it unless their JCL is modified to specify the IMSID or to use a STEPLIB.

    - Using different data sets with the same name on different MVS systems. For example, each MVS system′s catalog would point to its own copy of a library containing a DFSVC000 module. Each DFSVC000 module would point to the IMSID that was used on that MVS.

    - Using a system symbol as part of the data set name for started tasks. MVS/ESA SP V5 introduced system symbols. These symbols might have different values on different MVS systems, but can **only be used**

**with started tasks**, not jobs. BMPs will not use an IMS system which is moved from its home MVS system unless their JCL is modified to specify the IMSID.

The use of system symbols can be best explained with an illustration. The following example shows how a system symbol allows the same started task JCL to refer to different libraries on different systems. The &SYSNAME system symbol will have the name of the MVS system where it is used. The BMP′s STEPLIB would include a DD statement with the following:

```
DSN=IMS.IMSP.&SYSNAME.IMSID
```

On MVSA, this would be converted to:

```
DSN=IMS.IMSP.MVSA.IMSID
```

This data set would contain the DFSVC000 module for the IMS system run on MVSA. This module would have the MVS A′s IMS system as its default IMSID.

On MVSB, the DD statement would be converted to:

```
DSN=IMS.IMSP.MVSB.IMSID
```

This data set would contain the DFSVC000 module for the IMS system run on MVSB. This module would have the MVS B′s IMS system as its default IMSID.

This scheme would allow the same started task JCL to be used on different MVS systems with different results. On MVSA, the IMSID for its IMS system would be used. On MVSB, the IMSID for its IMS system would be used.

- The third method, using automated operations, uses replies to the DFS690A message.

  When a BMP attempts to connect to a control region and the IMS control regions identified in its IMSID and ALTID parameters are not available, message DFS690A message is sent to the MVS console. The text of the message is:

  ```
  DFS690A job.step.proc - CTL PGM NOT ACTIVE, REPLY ′WAIT′, ′CANCEL ′
  OR ′ALT-ID′
  ```

  If the operator replies with other than ′WAIT′, ′W′, ′CANCEL′, or ′C′, the BMP will retry the connect using the ALTID specified in the reply. An automated operations routine can be used to reply to the DFS690A message with the appropriate IMS control region for the MVS system on which the message is issued.

  If there are potentially multiple IMS control regions on the same MVS, the automated operations technique must know which is the correct one for the BMP.

- The fourth method, using the LOCAL parameter on the /START REGION command, is new in IMS/ESA 6.1.

  If BMPs are started with the IMS /START REGION command, the LOCAL parameter can be used. The command has the following form:

```
/START REGION membername LOCAL
```

When LOCAL is used, the IMSID found in the DFSVC000 member of the system on which the command is submitted is used for the job. This is true even if the IMSID is specified in the JCL for the job. That is, the LOCAL specification overrides what is coded in the JCL. Care must be used to ensure that the job is executed on the MVS system where the IMS system used for the command is running. This is true because the IMSID substitution is done as part of the command execution, not as part of the submitted job′s execution. JES2 installations can use a /*JOBPARM JCL statement to cause the job to be executed on the MVS system where the command is executed. This is done be specifying the SYSAFF=* parameter. The JCL statement would be:

```
/*JOBPARM SYSAFF=*
```

### 17.6.4 Maintenance Levels of RESLIB

The RESLIB used by a BMP does not have to be at the same maintenance level as the RESLIB used by its control region. This allows installations to apply maintenance to different control regions at different times. When doing this, an installation does not have to control the routing of a BMP to a control region with a matching maintenance level.

The RESLIB used by a BMP must be at the same release level as the control region it uses.

These considerations apply to all IMS dependent regions. Any dependent region must be at the same release level as its control region. On the other hand, dependent regions do not have to be at the same maintenance level as their control regions.

### 17.6.5 Routing BMP Jobs

BMP jobs can be routed to a system or set of systems by either of two methods. First, initiators can be used. A job will run only on a system which has an initiator with the appropriate class. Second, JES2 or JES3 control statements can be used.

#### 17.6.5.1 Using Initiators to Control Routing

Initiators can be used to control the systems on which a BMP job will execute. If an installation wishes to restrict the execution of a BMP to a set of systems, it should start initiators for the BMP job class only on this set of systems.

#### 17.6.5.2 Using JES2 /*JOBPARM Statement to Control Routing

The JES2 /*JOBPARM statement can be used to control the routing of a job. The SYSAFF parameter limits the systems on which the job can be run.

The following statement can be used to limit the job to the system on which it was submitted.

```
/*JOBPARM SYSAFF=*
```

The following statement can be used to limit the job to the systems in the list.

```
/*JOBPARM SYSAFF=(sysa,sysb,sysc,...)
```

The following statement allows the job to run on any system in the JES2 MAS.

```
/*JOBPARM SYSAFF=ANY
```

#### 17.6.5.3 Using JES3 //*MAIN Statement to Control Routing

The following statement can be used to limit the job to the systems in the list.

```
//*MAIN SYSTEM=(sysa,sysb,sysc,...)
```

The following statement allows the job to run on any system in the JES3 complex.

```
//*MAIN SYSTEM=ANY
```

# Chapter 18. Operations

Operating and supporting an IMS Parallel Sysplex data-sharing environment will require changes to a number of procedures currently in place and new procedures to manage the new facilities.

## 18.1 Operational Procedures

The following are examples of procedures which are used to manage the IMS operational environment.

- IMS system startup and shutdown
- IMS TM MTO or DBCTL console operations
- BMP restart
- Performance monitoring and reporting

## 18.2 Automated Operations

The cloning of an IMS system into multiple IMS systems will probably have an impact on an installation′s existing automation. The degree of impact will depend upon the function of the automation. If the automation function is “local” in nature (does not deal with shared or common resources), only minor changes would be required. However, if the automation function is “global” in nature (deals with shared or common resources), major changes could be required.

A common example of “local” automation would be to restart a transaction and program following an application program abend (message DFS554). Many installations have this type of automation implemented. Since an application program abend is an event that is localized to a particular IMS system, this type of automation should require little, if any, change.

A common example of a “global” automation would be to deallocate (/DBR) a set of databases from all of the IMS systems so that the set of databases can be processed by non-sharing batch. In this case, the automation would need to be changed, since the objective is to /DBR the databases from all of the IMS systems within the IMSplex.

Automated operations within an IMS system can be accomplished through several facilities. Some of these are explained below.

### 18.2.1 IMS Time-Control Operations

**TM Only**

This section applies only to IMS TM.

IMS Time Controlled Operations is driven by scripts written by the user. Some of these scripts can be dependent on functioning in a single IMS environment. For example, TCO can be used to invoke a transaction at some predetermined time. It might not be appropriate for the transaction to be submitted on every system.

The scripts for different IMS systems might need to be different. The current script must be examined and, if necessary, tailored for the different IMS systems.

Since the initial script has a fixed name (DFSTCF), different scripts for different IMS systems must be stored in different libraries concatenated from STEPLIB.

### 18.2.2 IMS AO Exits

The actions taken by the AO exits, such as DFSAOUE0 and DFSAOE00, might need to be different in different IMS systems. The exit routines and any accompanying AO programs must be examined and, if necessary, tailored for the different IMS systems. DFSAOUE0 is available only to IMS TM.

Since these exit routines have fixed names, different routines for different IMS systems must be stored in different libraries concatenated from STEPLIB.

### 18.2.3 Other AO Products

Other AO products, such as Netview, need to be reviewed to determine whether changes are necessary to accommodate the IMSplex environment.

## 18.3 Use of IRLM SCOPE=NODISCON

The specification of SCOPE=NODISCON for an IRLM can be used to reduce overhead and increase availability. SCOPE=NODISCON keeps an IRLM in its data-sharing group even when there are no IMS subsystems identified to it. NODISCON support was added to IMS V6 through APAR PQ01040.

When SCOPE=GLOBAL is used, an IRLM leaves its data-sharing group when no active IMS is identified to it. It disconnects from its lock structure and leaves its XCF group. It will reconnect to its lock structure and rejoin its XCF group when another IMS subsystem identifies to it. SCOPE=NODISCON eliminates these disconnections. The IRLM remains connected to the structure and in the XCF group. This has potential performance and availability benefits. The use of SCOPE=NODISCON is recommended.

There are several advantages in keeping an IRLM connected to its data-sharing group when its has no IMS subsystems connected to it. They include the following:

- Reduced overhead

  Overhead can be reduced in the following ways.

  - When an IRLM is used for IMS batch jobs but has no online IMS subsystem using it, an IRLM using SCOPE=GLOBAL can repeatedly join and leave the data-sharing group. If a batch data-sharing job step ends and no other batch data-sharing job steps or online systems are active on the IRLM, the IRLM disconnects from its lock structure and leaves its XCF group. When the next batch data-sharing job step begins, the IRLM must connect to the structure and join the XCF group. This can happen many times. The overhead of these repeated connections and joins can be significant.
  - When an IRLM disconnects from its lock structure, it can be the global lock manager for some entries in the lock table. This might occur even though no IMS systems are identified to this IRLM. The responsibility for

global lock management of these entries must be transferred to another IRLM. In some cases, the reassignment can use considerable resources.

The use of SCOPE=NODISCON avoids these overheads.

- Increased availability

  The use of NODISCON keeps all IRLMs in the data-sharing group. If one fails, the other IRLM(s) will remain in the group. Without SCOPE=NODISCON, a data-sharing group has an availability exposure. This is especially true for IRLMs supporting IMS batch data-sharing without active IMS online systems on two or more IRLMs.

  To understand this exposure, one must understand how IRLMs maintain information about identified IMS subsystems. When an IMS identifies to an IRLM, its IRLM passes information about the IMS to all of the IRLMs in the data-sharing group. When an IMS system fails, all IRLMs in the data-sharing group keep information about this IMS. So, all IRLMs know about all active and failed IMSs. If another IRLM joins the group, knowledge of all IMSs is also passed to it. When a failed IMS is recovered, information about the failed IMS is deleted from the IRLMs.

  The information about active and failed IMSs is used in DBRC authorization processing. When an IMS subsystem requests data-sharing authorization for a database, all IMS systems currently authorized for the database must be known to the IRLM used by the requesting IMS. This ensures that lock processing can be done properly. The other IMSs may be either active or failed.

  Consider what happens when all IRLMs leave the data-sharing group. This occurs with SCOPE=GLOBAL when an IRLM has no active IMSs identified to it. Of course, if an IRLM or its MVS system fails, it also leaves the data-sharing group. If there are no IRLMs in a data-sharing group, the information about failed IMSs is lost. The first IRLM to join the group (actually, the first to recreate it) will be unaware of any failed IMS subsystems. This means that no database authorizations can be done for databases authorized to failed IMSs before all failed IMSs have been recovered. Recovery for online systems is done either by emergency restart or XRF. Fast Database Recovery does not do this type of recovery since it does not release database authorizations. Recovery for batch update jobs is done either by dynamic batch backout or by the batch backout utility.

  SCOPE=NODISCON eliminates this type of outage. Information about failed IMS subsystems is kept in the IRLMs even when there are no active IMSs. This allows new IMS subsystems to gain authorizations for databases which are also authorized to failed subsystems.

### 18.3.1 Use of RDI Regions

Before the introduction of SCOPE=NODISCON support for the IRLM, some IMS installations used IMS RDI regions to provide similar benefits. RDI regions are executed using ′RDI′ as the region type. They are dummy regions that identify to the IRLM and do little else. They were used to ensure that an IRLM remained in the data-sharing group when all other IMS subsystems had terminated.

It is no longer necessary to use an RDI region to keep an IRLM in its data-sharing group. SCOPE=NODISCON can be used for this purpose.

## 18.4 Recovery Procedures

The following procedures are invoked to prepare for or perform resource recovery. More detail can be found in Chapter 20, “Recovery Procedures” on page 197, Chapter 14, “Automatic Restart Manager (ARM)” on page 133, and Chapter 19, “Fast Database Recovery (FDBR)” on page 189.

- Database recovery
- Batch application recovery
- IMS system recovery
- IRLM recovery
- Coupling Facility/structure recovery

## 18.5 Support Procedures

The following procedures are used in support of the IMS environment.

- Change control

  If the IMS systems are cloned, changes to one system should also be reflected in the others, except for split applications or databases.

- IMS maintenance

  A process for applying maintenance to one or more of multiple, cloned IMS systems must be established.

- IMS system definition

  If the IMS systems are cloned, changes to one system should also be reflected in the other(s).

- MFS format generation
- DBD, PSB, and ACB generation
- Security generation
- Online change

  The ″staging″ phase of online change must be reviewed and updated to reflect changes made to accommodate multiple cloned IMS systems.

- Database reorganization

  Database reorganization procedures will require the database to be made unavailable to all IMS systems.

- Application maintenance

  Application code changes, and updates to application PGMLIBs, must be coordinated across multiple IMS systems.

- Coupling Facility structure size changes

  A procedure should be developed for changing Coupling Facility structure sizes. See 15.2, “Changing CF Structure Sizes” on page 151 for more information.

## 18.6 IMS Log Management

In a single IMS environment, there were only a single set of logs for IMS. In a multi-IMS environment, there are multiple sets of logs to be managed. In most cases, they are independent and (except for their data set names) can be managed independently. There are, however, a few considerations.

### 18.6.1 IMS Log Archive

In a single IMS environment, there is only a single set of logs. In a cloned environment, each IMS subsystem has its on set of logs. An installation can manage each set independently.

The IMS Log Archive utility must be run to archive OLDS. In a cloned environment, each IMS online system will have its own archive process. Log archive is run as a batch job, and it can run on any MVS system. It does not have to run on the same MVS system with the online system whose OLDS it archives.

When automatic archiving is used, the online system issues an internal GENJCL.ARCHIVE command. This selects the OLDS needing to be archived and creates a job for them. The job can be passed to the internal reader so that it will be executed without requiring any operator actions. This technique is also appropriate for data-sharing.

The class for an archive job is specified in the JOBJCL member of the DBRC skeletal JCL. It is reasonable for all archive jobs to use the same class; therefore, they can use the same skeletal JCL library. The number of initiators for this class might have to be increased as logging volumes grow.

The ARCHIVE member of the skeletal JCL will contain the JCL used for archive jobs. Since the subsystem name (SSID) is available for the GENJCL process, different systems can create output data sets which include their own SSID as a qualification in the data set name. The default ARCHIVE member uses such a name.

## 18.7 Job Scheduling Procedures

With multiple systems available, job scheduling procedures and JOB classes should be reviewed.

An installation might want to restrict BMPs to a particular system for performance or capacity reasons. This can be done with JOB classes. If an installation wants to run BMPs on multiple IMS systems, it must ensure that the IMSID parameters for the BMPs are correct. When a BMP is run, its IMSID parameter must specify the IMS system which it will use. Moving a BMP from one system to another will require that the IMSID change. See 4.3, “IMS Data-Sharing Subsystems” on page 22 for further information on these considerations.

The fix to APAR PQ21039 (see 17.6.1, “Using the Function Delivered by APAR PQ21039” on page 173) simplifies the specification for users whose BMPs can execute under multiple control regions in the IMSplex. With this enhancement, no changes to BMP JCL are required even when different control regions are used.

## 18.8 Online Change Procedures

Online change for each IMS online system is executed independently. For cloned systems, procedures will have to be developed to coordinate online change on all of the systems.

If the systems share the data set (ACBLIB, MATRIX, MODBLKS, or FORMAT) which is being switched, the coordination is simpler. In this case, the library is copied from the staging library to the inactive library by the Online Change utility (DFSUOCU0). The MODSTAT data set of any of the sharing subsystems can be used. Of course, they all must indicate the same DDNAME for the active library. /MODIFY PREPARE and /MODIFY COMMIT commands are issued independently for the online systems.

If each system has a unique version of the data set (ACBLIB, MATRIX, MODBLKS, or FORMAT) which is being switched, the Online Change utility must be run for each system. In this case, it is advisable for each invocation of the utility to refer to the MODSTAT data set associated with the system for which the copy is being done. /MODIFY PREPARE and /MODIFY COMMIT commands are issued independently for the online systems.

In either case, it is important for the change to be made in each of the cloned systems or for all to revert to their original library. For example, all should be using ACBLIBA or all should be using ACBLIBB when the procedures are completed.

Enqueues are used to protect libraries used with online change. The online systems always have shared enqueues on the active libraries. The Online Change utility holds exclusive enqueues on both the staging and target libraries. For example, if online system IMS1 is using ACBLIBA as its active library, it will hold a shared enqueue on ACBLIBA. If the Online Change utility is run using IMS1′s MODSTAT, it will select ACBLIBB as its target. The utility will request exclusive enqueues on the staging library, ACBLIB, and on the target library, ACBLIBB. Obviously, the Online Change utility cannot acquire the exclusive enqueue on ACBLIBB if another online system is using it for its active library. The enqueues would conflict.

## 18.9 IMS Commands from MVS Consoles

IMS/ESA V6.1 includes the following support for IMS commands issued from MVS consoles.

- Command Recognition Characters (CRC) use with IMS TM

  Previous releases of IMS allowed CRC use only with DBCTL. IMS V6.1 allows the use of CRCs with both DBCTL and IMS TM. For example, if # is the CRC, the following display command can be entered from an MVS console with either IMS TM or DBCTL.

  ```
  #DISPLAY ACTIVE
  ```

- Use of the IMS subsystem name for IMS TM command entry from MVS consoles

  Previous releases of IMS allowed the use of the subsystem name only for DBCTL commands. IMS V6.1 allows the use of the IMS subsystem name with commands for both DBCTL and IMS TM. For example, if IMS1 is the IMS subsystem name, the following display command can be entered from an MVS console with either IMS TM or DBCTL.

```
IMS1DISPLAY ACTIVE
```

- Use of the same CRC by multiple IMSs in the same MVS

  Previous releases of IMS did not allow multiple DBCTLs on the same MVS to use the same CRC. IMS V6.1 allows multiple DBCTL or IMS TM systems to use the same CRC. When two subsystems on the same MVS have the same CRC, any command entered using this CRC will be processed by both subsystems.

- Support for IMS TM and DBCTL commands from MCS and E-MCS consoles

  Previous releases of IMS allowed IMS TM to use only WTOR outstanding replies for entering commands from MVS consoles. IMS V6.1 allows both IMS TM and DBCTL to enter commands by using a CRC or the subsystem name. Responses to commands entered from MCS and E-MCS consoles are returned to the originating console. This includes both synchronous and asynchronous responses.

These facilities allow commands for multiple IMS systems on different MVSs in a Sysplex to be entered from one console. The MVS ROUTE command can be used to send IMS commands to any MVS system in a Sysplex.

The following illustrates the use of the ROUTE command in this environment. Assume that IMS1 is running on MVS1, IMS2 is running on MVS2, and IMS3 is running of MVS3. The Command Recognition Character is # for all of the IMS systems.

The following DISPLAY command would be processed by IMS1 on MVS1.

```
ROUTE MVS1,IMS1DISPLAY ACTIVE
```

The following DISPLAY command also would be processed by IMS1 on MVS1.

```
ROUTE *ALL,IMS1DISPLAY ACTIVE
```

The following DISPLAY command would be processed by all IMS systems.

```
ROUTE *ALL,#DISPLAY ACTIVE
```

### 18.9.1 GLOBAL Commands

Block-level data-sharing allows the use of the GLOBAL parameter on the following commands.

```
/START DB
/STOP DB
/DBD DB
/DBR DB
/START AREA
/STOP AREA
/DBR AREA
```

The GLOBAL parameter causes the command to be sent to other IMS online subsystems in the data-sharing group. For example, database DI21PART can be started on all online IMS subsystems by issuing the following command.

```
/START DB DI21PART GLOBAL
```

When GLOBAL commands are processed on other IMS systems, their responses are not sent to the entering terminal or console. For IMS TM, the responses are sent to the processing system′s IMS Master terminal. For DBCTL, the responses are sent to the MVS console on the processing system. This is often inconvenient. For example, if a BMP with intent against the database is executing on any system, a /DBR command will be rejected on that system. The following message will be issued on that system.

```
DFS0565I /DBR COMMAND NOT PROCESSED DB=xxxxxxxx IN USE BY
             PSB=psbname REG=region-number
```

**Note:** /DBR DB commands for DEDBs are processed immediately even if the DEDBs are scheduled by a BMP.

GLOBAL commands can also affect DBRC flags. If the NOPFA parameter is not also specified, the following occur.

- Some commands set the DBRC ″prohibit further authorization″ flag for the database or area. This does not occur without the GLOBAL parameter.

  The /STOP DATABASE, /STOP AREA, /DBR DATABASE, and /DBR AREA commands set the ″prohibit further authorization″ flag. This prevents authorization by any user other than a database reorganization, image copy, or recovery utility. For example, an IMS batch job could not be authorized to use the database.

- Some commands set the DBRC ″read only″ flag for full function databases. This does not occur without the GLOBAL parameter.

  The /DBD DATABASE command sets the ″read only″ flag. This prevents authorization to subsystems with update or exclusive intent. Subsystems requesting read or read-only intent, Image Copy, Online Image Copy, Scan, and Surveyor utilities can be authorized.

- The /START DATABASE, /START AREA, and /DBD DATABASE commands reset the ″prohibit further authorization″ flag.
- The /START DATABASE and /DBR DATABASE commands reset the ″read only″ flag.

When NOPFA is specified with GLOBAL on the /DBR, /DBD, and /STOP commands, the DBRC ″prohibit further authorization″ and ″read only″ flags are not affected.

The GLOBAL parameter has other restrictions.

- There is a limit of 29 databases or areas that can be specified on a GLOBAL command. Message DFS3327 is issued when the limit is exceeded.
- The DATAGROUP parameter cannot be used with the GLOBAL parameter. The DATAGROUP parameter was introduced in IMS/ESA V5.1. It allows an installation to define a group of databases in DBRC. This group′s name can be used in database commands without the GLOBAL parameter. When DATAGROUP is used, the command is issued for each member of the group.

### 18.9.2 Alternative to GLOBAL Commands

Many installations might find the use of the ROUTE *ALL command from an MVS console to be more convenient than the use of GLOBAL commands. It has the following advantages.

- All responses from all IMS systems are sent to the originating console.
- All commands are supported. The GLOBAL parameter is supported only on /START, /STOP, /DBR, and /DBD commands for databases and areas. It is not supported for other commands, such as /DISPLAY DB.
- DBRC flags are not affected. The GLOBAL parameter also requires the NOPFA parameter to avoid setting or resetting the ″prevent further authorization″ or ″read only″ flags in DBRC.

### 18.9.3 Recommendations for IMS Commands From MVS Consoles

The following scheme allows for all IMS commands to be entered from a single MCS or E-MCS console. A command can be directed either to all IMS systems in the sharing group or to a particular IMS.

- Use the same CRC for all IMS subsystems.
- Use the ROUTE *ALL command with the CRC to send a command to all IMS subsystems. Use this capability instead of GLOBAL commands.
- Use the ROUTE command with the MVS name and the IMS subsystem name to send a command to one IMS subsystem. Alternatively, *ALL can be used instead of the MVS name; however, this unnecessarily sends the command to all MVS systems in the sysplex.

IMS subsystems can be moved between MVS systems. This is especially important in recovery scenarios where it might be necessary to restart an IMS subsystem on a different MVS.

# Chapter 19. Fast Database Recovery (FDBR)

Fast Database Recovery (FDBR) is an optional function of IMS introduced in IMS/ESA Version 6. It is designed to quickly release locks held by a IMS subsystem when the subsystem fails. FDBR monitors an IMS subsystem using the IMS log and XCF status monitoring services. When it determines that the IMS has failed, FDBR dynamically recovers databases by backing out uncommitted updates for full-function databases and executing redo processing for DEDBs. These are the functions that would be performed by emergency restart. When FDBR completes the back outs and redos, it releases the locks held by the failed subsystem. This allows any sharing subsystems to acquire these locks. From the time of the IMS failure until the locks are released, other subsystems requesting the locks will receive lock rejects. Lock rejects usually result in U3303 abends of application programs.

FDBR is not a replacement for emergency restarts of IMS. Emergency restarts are still required to do the following:

- Recover the message queues
- Recover MSDBs
- Resolve in-doubt threads with CICS and DB2
- Change the DBRC subsystem record from ′failed′ to ′normal′
- Release DBRC authorizations

The benefit of FDBR is its quick release of in-flight locks. This lessens the likelihood of lock rejects in other IMSs.

## 19.1 FDBR Monitoring

FDBR uses two techniques to monitor the IMS it is tracking. It reads the log being written by IMS, and it uses XCF status monitoring.

### 19.1.1 Log Monitoring

Each second, FDBR reads the IMS log until it reaches logical end-of-file (EOF). If IMS abends, it writes a x′0607′ log record. When FDBR reads this record, it invokes its recovery actions and sends the following message:

```
DFS4166I FDR FOR imsid DB RECOVERY PROCESS STARTED,REASON = IMS FAILURE
```

FDBR also has a log timeout function. If IMS does not write any log records for five seconds, FDBR writes a warning message. However, the lack of a log record will not cause FDBR to invoke its recovery processing. The warning message is:

```
DFS4164W  FDR FOR imsid DETECTED TIMEOUT DURING LOG AND XCF SURVEILLANCE
```

While reading the log, FDBR maintains copies of some database-related control blocks and buffers. This is similar to the processing that emergency restart does when reading the log. These control blocks and buffers are used for recovery processing.

## 19.1.2 XCF Status Monitoring

When FDBR is used, IMS and its tracking FDBR system join an XCF group. XCF status monitoring is used to inform FDBR if the IMS system has failed. XCF monitors both the status of the IMS system and the status of the MVS on which it is executing.

### 19.1.2.1 XCF Monitoring of IMS

If the IMS system fails to update its status field within the user-defined timeout interval, FDBR is notified by its XCF Group User routine. In this case, FDBR writes a warning message:

```
DFS4165W  FDR FOR imsid XCF DETECTED TIMEOUT ON ACTIVE IMS SYSTEM, REASON = IMS SURV.
          DIAGINFO = xxxx
```

As with log timeouts, FDBR will not invoke recovery for an XCF-detected timeout.

If the IMS system's address space terminates, FDBR is notified by its XCF Group User routine. In this case, FDBR writes a message and invokes its recovery processing. The message is:

```
DFS4166I  FDR FOR imsid DB RECOVERY PROCESS STARTED, REASON = XCF NOTIFICATION
```

Note that this only occurs when the address space terminates. It is not invoked for merely an abend. If IMS is a started task, an abend will cause address space termination. If IMS is a job, an abend will not cause address space termination. So, XCF status monitoring of the IMS system will cause automatic invocation of recovery processing for IMS started tasks, but not for jobs.

### 19.1.2.2 XCF Monitoring of MVS

If the MVS system fails to update its Sysplex couple data set within the user-defined timeout interval, FDBR is notified by its XCF Group User routine. FDBR writes a warning message. The message is:

```
DFS4165W  FDR FOR imsid XCF DETECTED TIMEOUT ON ACTIVE IMS SYSTEM,
          REASON = SYSTEM, DIAGINFO = xxxx
```

This user-defined timeout interval is specified by the TIMEOUT parameter in the IMS DFSFDRxx member. This timeout does not cause FDBR to invoke its recovery processing.

A failure of the MVS system to update its Sysplex couple data set might result in system isolation. System isolation causes the MVS system to be removed from the Sysplex. Its I/Os and Coupling Facility accesses are terminated, its channel paths are reset, and a non-restartable wait state is loaded. The invocation of system isolation is controlled by the ISOLATETIME parameter in the Sysplex Failure Management (SFM) policy for the Parallel Sysplex. If the Sysplex couple data set is not updated in the specified time interval, system isolation is invoked. If system isolation occurs for the MVS on which IMS is running, FDBR is notified by its XCF Group User Routine. This causes FDBR to write a message and invoke its recovery processing. The message is:

```
DFS4166I  FDR FOR imsid DB RECOVERY PROCESS STARTED, REASON = XCF NOTIFICATION
```

## 19.2 Invoking Recovery

There are two parts to FDBR recovery. First, FDBR must complete reading all of the log records to prepare for the full-function backouts and DEDB redos. Second, it must do the backouts and redos. Completion of the log reads can be done as soon as FDBR is aware that IMS is no longer running. This is signaled by one of three methods:

- FDBR reads an IMS abend log record (type x′0603′).
- XCF invokes the FDBR Group User Routine informing FDBR of the termination of MVS or the IMS address space.
- An operator command is issued requesting an FDBR recovery. This command is:

```
F fdbrproc,RECOVER
```

FDBR cannot begin the backouts and redos before it is sure that all I/O which the failed IMS has started is complete. MVS has a function that indicates that this I/O is complete. It is called I/O prevention. When I/O prevention is complete for an address space, MVS has ensured that no more I/Os will be done by it. Of course, if MVS fails, it cannot invoke I/O prevention. In this case, the system isolation function of Parallel Sysplex ensures that no more I/Os are being done. Finally, the termination of an address space indicates that all I/Os are complete. XCF monitoring will make FDBR aware of the system isolation of an MVS or address space termination. In other cases, the operator must make FDBR aware of the complete of all I/Os. When the operator issues the `F fdbrproc,RECOVER` command, FDBR assumes that I/Os by the failed IMS have completed.

In cases where FDBR is not aware that the failed IMS′s I/Os have completed, it issues the following command during recovery processing:

```
DFS4167A FDR FOR imsid WAITING FOR ACTIVE SYSTEM TO COMPLETE I/O PREVENTION.
         REPLY ″UNLOCK″ WHEN I/O PREVENTION COMPLETES.
```

The operator should reply ″UNLOCK″ only after the ensuring that I/O prevention is complete. The MVS AVM006A messages indicates this. It is issued by the MVS system on which IMS was running. The message is:

```
AVM006A  TELL OPERATOR AT BACKUP TO REPLY ′UNLOCK′ TO MESSAGE AVM005A.
         I/O PREVENTION IS COMPLETE FOR SUBSYSTEM imsid,
         FAILING ACTIVE ELEMENT OF RSE imsid.
```

If system isolation or address space termination completes after the DFS4167A message is issued, the operator does not need to reply. In these cases, FDBR is made aware of the situation through its XCF Group User Routine. FDBR clears the DFS4167A message and issues the following message:

```
DFS4171I FDR FOR imsid ACTIVE IMS TERMINATION NOTIFIED BY XCF.  OPERATION RESUMED.
```

## 19.3 FDBR Failures

If FDBR fails while it is tracking IMS, it can be restarted and resume tracking. The restart can be done by the operator or by the Automatic Restart Manager (ARM). ARM support for FDBR is explained below.

If FDBR fails while it is in database recovery phase, it cannot be restarted. FDBR restart requires that its associated IMS be active. If FDBR fails in its database recovery phase, the IMS system should be emergency restarted. This will complete the database recovery processes.

## 19.4 Restarting IMS after FDBR Completion

IMS must be restarted after FDBR completes its recovery processes. Restart is required to resolve in-doubt threads with CICS and DB2, recover message queues, change the DBRC subsystem record from failed to normal, release DBRC authorizations, and recover MSDBs. This recovery should be done as soon as possible, but cannot be done before FDBR completes. Of course, an installation will also want the restarted IMS to process new work.

When FDBR completes its recovery processing, it issues the following message:

```
DFS4168I FDR FOR imsid DATABASE RECOVERY COMPLETED.
```

Automation can use this message to indicate that FDBR′s recovery is complete and that the associated IMS should be emergency restarted.

## 19.5 DBRC Authorizations with FDBR

FDBR does not change the DBRC subsystem record from failed to normal. It also does not release database authorizations. When FDBR completes, it has released locks held by the failed IMS, but this IMS still holds database authorizations. Other IMS subsystems cannot obtain new authorizations for databases which are authorized to the failed IMS. Authorization protocols prevent this by checking that all authorized subsystems are known to the IRLMs before granting new authorizations for a database. This implies that from the time FDBR completes its recovery until the IMS subsystem is emergency restarted, new database authorizations will not succeed for any databases authorized to the failed IMS. Restarting the failed IMS quickly is highly recommended.

## 19.6 ARM and FDBR

ARM support is provided for both IMS and FDBR.

### 19.6.1 ARM Support for IMS With FDBR Active

ARM support is available for IMS through the ARMRST=Y execution parameter. See Chapter 14, “Automatic Restart Manager (ARM)” on page 133 for information on this support without FDBR. When an FDBR system using ARM tracks an IMS system, it notifies ARM that it is doing the tracking and that ARM should not restart this IMS if it fails. FDBR uses the ASSOCIATE function of ARM to make this notification. So, even if IMS registers to ARM, IMS will not be restarted after its failures when FDBR is active and using ARM.

If FDBR is terminated normally, it notifies ARM. This tells ARM to restart the tracked IMS if this IMS has previously registered to ARM. This is appropriate since FDBR is no longer available to do the recovery processes.

When ARM is used for IMS, an installation can choose either to use or not to use ARM for FDBR. If FDBR does not use ARM, it cannot tell ARM not to restart IMS. In this case, a failure of IMS will cause FDBR to do its processing and ARM to restart IMS. This could be advantageous. One would expect FDBR processing to complete before the restart of IMS by ARM completes. If so, locks would be released quickly by FDBR and restart of IMS by ARM would occur automatically. Since these actions depend on the timing of FDBR processing and IMS restart processing, they cannot be guaranteed.

### 19.6.2 ARM Support for FDBR

FDBR can register to ARM. This is controlled by the ARMRST execution parameter. The default is to register. ARMRST=N causes FDBR not to register.

When FDBR registers to ARM, it is restarted automatically if it fails. Failures include abends of FDBR and failures of the MVS system on which it is executing.

When IMS registers with ARM, it uses its IMSID for its ARM element name. Similarly, when FDBR registers with ARM, it uses its IMSID for its ARM element name. Since ARM element names must be unique, the IMSIDs of an IMS system and an FDBR region cannot be the same.

## 19.7 FDBR, XRF, and ARM

FDBR, XRF, and ARM can be used to back out the in-flight work of a failed IMS system and release its locks. Each has its own advantages.

### 19.7.1 FDBR Advantages and Disadvantages

FDBR has the following advantages:

- FDBR begins its backout processing sooner than an IMS restart initiated by ARM. FDBR is tracking the active IMS by reading its log. This eliminates most of the log reading that is required in an ARM-initiated restart.
- FDBR requires no operator intervention when IMS runs as a started task. XCF monitoring is used to inform FDBR that IO prevention is complete.
- FDBR uses less CPU and storage than XRF during the tracking phase.

FDBR has the following disadvantages:

- FDBR does not restart the failed IMS.
- FDBR does not invoke the following recovery processes:
  - Resolution of in-doubt threads with CICS and DB2
  - Recovery of IMS message queues
  - Recovery of MSDBs
  - Release of DBRC database authorizations

### 19.7.2 XRF Advantages and Disadvantages

XRF has the following advantages:

- XRF begins its database recovery processing sooner than an IMS restart initiated by ARM. XRF is tracking the active IMS by reading its log. This eliminates most of the log reading that is required in an ARM-initiated restart.
- XRF can resolve in-doubt threads with CICS and DB2. This requires that the CICS and DB2 subsystems be restarted on the MVS where the XRF alternate executes.
- XRF handles the recovery of IMS message queues.
- XRF moves terminal sessions to the new active system.
- XRF recovers MSDBs.
- XRF accepts new work on the new active system.
- XRF handles DBRC database authorizations by having the alternate assume the authorizations of the failed active

XRF has the following disadvantages:

- XRF requires more CPU and storage than FDBR while in tracking phase.
- Takeover on another CPU requires operator intervention to indicate that IO prevention is complete.

### 19.7.3 ARM Advantages and Disadvantages

ARM has the following advantages:

- ARM requires no resources for tracking.
- ARM-initiated restarts can resolve in-doubt threads with CICS and DB2. ARM can automatically move IMS, CICS, and DB2 in a group to the same MVS.
- ARM-initiated restarts do not require operator intervention.
- ARM-initiated restarts handle the recovery of IMS message queues.
- ARM-initiated restarts recover MSDBs.
- ARM-initiated restarts accept new work on the new active system.
- ARM-initiated restarts release DBRC database authorizations during emergency restart processing.

ARM has the following disadvantages:

- ARM-initiated restarts begin the database recovery processes later than FDBR or XRF would.

## 19.8 Recommendations

The following actions are highly recommended to make FDBR most effective. If they are not followed, some FDBR recoveries might require operator actions.

- Execute IMS as a started task, not a job.

If IMS is executed as a job, abends require operator action on the FDBR system to complete recovery processing. The operator must manually inform FDBR of the completion of I/O prevention.

- Specify ISOLATETIME in the Sysplex Failure Management (SFM) policy.

  This causes system isolation to be invoked for a failed MVS system. A specification of PROMPT, instead of ISOLATETIME, requires operator actions when XCF status monitoring recognizes the time out of an MVS system.

- Use a name that associates an FDBR with the IMS it tracks.

  Commands for FDBR are issued by using an MVS modify command and specifying the FDBR procname. This procname should be easily associated with the IMS system which the FDBR system tracks. For example, the FDBR procname could include the IMSID. Similarly, the XCF group name used for FDBR should also include the IMSID. The default group name is ″FDR″ followed by the IMSID. It is reasonable to make the FDBR procname the same as the XCF group name.

# Chapter 20. Recovery Procedures

We discuss some possible recovery procedures here.

## 20.1 Image Copies

IMS has four methods of image copying databases. These are:

- Database Image Copy (DFSUDMP0)
- Concurrent Image Copy (DFSUDMP0 with CIC option)
- Online Database Image Copy (DFSUICP0)
- Database Image Copy 2 (DFSUDMT0)

### 20.1.1 Database Image Copy (DFSUDMP0)

The Database Image Copy utility without the CIC parameter produces clean image copies. That is, these copies are not made while updates are being made to the database. DBRC authorizations enforce this. Subsystems might not have update authorization for a database when one of its data sets is being image-copied in this manner.

This image copy can be used to back up HDAM, HIDAM, HISAM, SHISAM, secondary index, and DEDB database data sets.

### 20.1.2 Concurrent Image Copy (DFSUDMP0 with CIC Option)

The Database Image Copy utility can be executed with the CIC parameter. This allows the utility to copy a database data set while other subsystems have update authority to update the database. When this is done, a fuzzy image copy is produced. A subsequent recovery of the database requires both the image copy and the update log records for its inputs. Concurrent Image Copies require the use of DBRC with the database registered as SHARELVL 1, 2, or 3.

KSDSs cannot be copied by Concurrent Image Copy. This restriction is implemented to avoid a concurrent CI/CA split which could move a logical record into a CI that is never copied by the utility.

This image copy can be used to back up HDAM, HIDAM, HISAM overflow, and DEDB database data sets. It will not back up a HIDAM index.

### 20.1.3 Online Database Image Copy (DFSUICP0)

Online Image Copy (OLIC) runs as a special BMP in an online system. It produces a fuzzy image copy. Concurrent updates in the same online system are allowed, but other subsystems cannot have update authorization for the database. OLIC acquires its own database authorization from DBRC to enforce this rule.

OLIC can be used to back up HDAM, HIDAM, HISAM, SHISAM, and secondary index databases data sets. It can not be used for DEDBs.

### 20.1.4 Database Image Copy 2 (DFSUDMT0)

Database Image Copy 2 can be used to produce either a clean image copy or a fuzzy one. Since this utility invokes DFSMS concurrent copy to dump the data set, it requires that the data set reside on a storage control unit that supports this DFSMS function. A clean image copy requires that the database or area not be authorized for update to any IMS subsystem during the logical copy operation. Logical copies typically take only a few seconds. Fuzzy image copies can be done while the database or area is authorized for update.

If a KSDS data set is copied while it is being updated, BWO(TYPEIMS) must be specified on the AMS DEFINE or ALTER. If BWO(TYPEIMS) is not specified for a KSDS, its database must be stopped before executing the utility. That is, BWO(TYPEIMS) is required for fuzzy copies of KSDSs.

This image copy can be used to back up HDAM, HIDAM, HISAM, SHISAM, secondary index, and DEDB database data sets.

### 20.1.5 Summary of Image Copies

Table 9 table summarizes the restrictions for concurrent updates when using each of the image copy types.

*Table 9. Image Copy With Concurrent Updates*

| Image Copy Type | Restrictions |
|---|---|
| Image Copy (DFSUDMP0) without CIC | Concurrent updates not allowed |
| Image Copy (DFSUDMP0) with CIC | Concurrent updates of KSDS not allowed |
| Online Image Copy (DFSUICP0) | Concurrent updates by other subsystems not allowed |
| Image Copy 2 (DFSUDMT0) | None |

## 20.2 Database Recoveries

- Change Accumulation must be used for shared databases.
  - GENJCL.CA can be used to generate JCL.
  - GENJCL.RECOV generates JCL using CA input.

    If a CA is required but has not been done, GENJCL.RECOV will produce a message indicating the CA must be done before the recovery can be done.

  It is not necessary to execute Change Accumulation as a regular process. Instead, it can be run only when a database recovery is required. Of course, this will affect how quickly a database can be recovered.

- The database must be deallocated from all systems before the recovery can be done. Operational procedures for issuing the appropriate /DBR commands must be established. This can be done by issuing the commands on each IMS subsystem, by issuing a /DBR command with the GLOBAL parameter, or by using an MVS ROUTE command to send the /DBR command to each IMS subsystem. See 18.9, “IMS Commands from MVS Consoles” on page 184 for a discussion of these alternatives. Operators should verify that the database has been deallocated on each sharing system.

After the recovery, the database should be started on all sharing systems. This can be done by issuing /START commands on each IMS subsystem, by issuing a /START command with the GLOBAL parameter, or by using an MVS ROUTE command to send the /START command to each IMS subsystem. See 18.9, “IMS Commands from MVS Consoles” on page 184 for a discussion of these alternatives.

### 20.2.1 Time-Stamp Recovery Considerations

A time-stamp recovery is a database recovery action that recovers an IMS database to a prior state or time-stamp, rather than its most current state. Hence, the name, *time-stamp* recovery.

Normal IMS facilities dictate that the database must not have been in use by any IMS subsystem at the desired prior state or time-stamp. In other words, an IMS *recovery point* must have been established prior to the need for the IMS *recovery point*. An IMS recovery point is normally established by issuing /DBR or /DBD commands (without the NOFEOV parameter) for the database or databases against all of the IMS sharing subsystems, waiting for all of the commands to successfully complete, and then issuing /STA commands for the database or databases against all of the IMS sharing subsystems. It is imperative that the /DBR or /DBD commands complete on all of the IMS subsystems prior to the issuance of /STA commands on any of the IMS subsystems.

The establishment of IMS recovery points will result in a loss of database availability (either complete loss of availability in the case of /DBR commands or loss of update capability in the case of /DBD commands) while the recovery point is being established. In addition, the establishment of IMS recovery points requires the issuance of multiple, coordinated commands across multiple IMS subsystems.

If the establishment of IMS recovery points is a problem or concern to the user, the use of a product such as IMS Recovery Saver (IBM product number 5565-A68) should be considered. IMS Recovery Saver allows IMS databases to be recovered to any time-stamp without requiring the existence of an IMS recovery point.

The high level process, when using IMS Recovery Saver to perform a time-stamp recovery, is:

1. Create a backup of the *production* RECON.
2. Restore the backup copy of the RECON.
3. Execute IMS Recovery Saver (specifying the desired time-stamp) using the *backup* RECON.
4. Recover the desired databases.
5. Perform all necessary backout operations.
6. Inform the *production* RECON that the recoveries have been performed.
7. Take image copies of the desired databases.

For additional information, see *IMS/ESA Recovery Saver User′s Guide and Reference*, SC26-9191.

## 20.3 IMS Batch (DBB and DLI) Job Abends

Procedure required for block-level data-sharing batch jobs.

- ARM can not be used.
- Backout any update job using the Batch Backout utility.

  The utility must have a CFNAMES statement in its DFSVSAMP data set.

## 20.4 IMS Online (TM or DBCTL) Abends

Procedure required for each online system.

- ARM can be used.
- Same procedures as without data-sharing.

  Lock rejects can cause other systems to be affected until recovery is completed.

## 20.5 IRLM Abends

Procedure required for each IRLM.

- Start IRLM
- For IMS online systems:
  - ARM can be used.
  - MVS console: issue the F imsproc,RECONNECT command.

    This command might not be required. IMS internally issues this command after quiescing. If the IRLM has been restarted before IMS quiesce completes, the user does not have to issue this command. If ARM is used to restart IRLM, IRLM restart should complete before IMS quiesce completes.
  - Issue the /START AREA command for all DEDB areas.

    Can use Datagroup, such as the following:

    ```
    /START DATAGROUP(ALLAREAS)
    ```
- For batch (DLI and DBB) data-sharing jobs:
  - Back out update jobs.

    The utility must have a CFNAMES statement in its DFSVSAMP data set.
  - Restart all jobs.

## 20.6 MVS Failures

Procedure required for each MVS.

- Restart MVS (if possible)
- Restart IRLM (if MVS restarted)
- Emergency restart IMS online (TM or DBCTL)

  ARM can be used.

This can be done on any MVS using any IRLM in the data-sharing group. If the restart uses a different IRLM with a different IRLM name, the IRLM name specified by IMS must be changed.

The restart is done to invoke dynamic backout for in-flight work. If the IMS system is moved to a different MVS, it is not necessary to do any new work on this IMS, unless the restart of BMPs requires that they use the same IMS system.

- Restart BMPs.

  If LAST is specified for CKPTID, the same IMS system must be used. If the restart records are read from the IMS log, the same IMS system must be used.

  If the restart is done by application logic using data in an IMS database, the restart can be done on any IMS system in the data-sharing group. If the restart records are read using an IMSLOGR DD statement, the restart can be done on any IMS system in the data sharing group.

- Emergency restart CICS (if used).

  ARM can be used.

  This is needed to resolve any in-doubts between CICS and DBCTL.

- Restart DB2 (if used).

  ARM can be used for DB2 V4 or later releases.

  This is needed to resolve any in-doubts between IMS TM and DB2.

- Batch (DLI and DBB) data-sharing jobs:

  – Back out update jobs and any read jobs which hold modify locks.

    The utility must have a CFNAMES statement in its DFSVSAMP data set.

  – Restart all jobs.

## 20.7 Lock Structure Failures

Automatic rebuild of structure is attempted.

IMS batch (DLI and DBB) data-sharing jobs abend. Backout is required for updaters. All batch data-sharing jobs must be restarted. Procedure for rebuild failure.

- Correct the cause of failure. Possibilities include:

  – Failure of the CF containing the structure with no other available CF in the structure′s preference list.

  – Failure of the CF containing the structure and insufficient storage on CFs in the preference list.

  The correction can be a redefinition of the the policy to provide another CF in the preference list. It can be a repair of a CF or the creation of one in an LPAR.

- Issue the SETXCF REBUILD command.

```
SETXCF START,REBUILD,STRNAME=LOCKIRLMP01
```

This will cause the structure to be rebuilt. The IMS systems will automatically reconnect to the IRLMs, and the IRLMs will automatically reconnect to the structure.

## 20.8 OSAM and VSAM Structure Failures

Automatic rebuild of structure is attempted.

All buffers are invalidated (OSAM or VSAM).

IMS batch (DLI and DBB) data-sharing jobs abend. Backout is required for updaters. All batch data-sharing jobs must be restarted. Some online transactions might abend.

Procedure for rebuild failure

- IMS will issue message:

  ```
  DFS3384I DATA SHARING STOPPED.
  ```

  All SHARELVL=1, 2, and 3 full-function databases are stopped. This includes both OSAM and VSAM databases, even if only one of these structures is lost.

- Correct the cause of failure. Possibilities include:

  – Failure of the CF containing the structure with no other available CF in the structure′s preference list.

  – Failure of the CF containing the structure and insufficient storage on CFs in the preference list.

  The correction can be a redefinition of the the policy to provide another CF in the preference list. It can be a repair of a CF or the creation of one in an LPAR.

- Issue SETXCF REBUILD command.

  ```
  SETXCF START,REBUILD,STRNAME=CACHIMSVP01
  ```

  This will cause the structure to be rebuilt. The IMS systems will automatically reconnect to the structures.

## 20.9 DEDB VSO Structure Failures

Automatic rebuild of these structures is not attempted.

If there are two structures for the area and the second one remains available, processing continues with the remaining structure.

If the failed structure is the only remaining structure for the area, the area is stopped on each system. Committed changed data might not have been written to DASD. DBRC sets the recovery needed flag for the area.

### 20.9.1 Procedure for Failure of One of Two Structures

If only one of two structures for an area fails, it is not necessary to recover immediately. Processing continues with the remaining structure.

The use of two structures can be reinstituted after all IMS subsystems discontinue using the current structure. This occurs when they close the area. The next time the area is opened, connections to the two structures will be attempted. This will cause the structures to be built. If the builds are successful, the use of two structures is reestablished. It might be necessary to change the CFRM policy so that the structures can be built on available coupling facilities.

### 20.9.2 Procedure for Failure of Only Structure or Both Structures

If the only remaining structure for an area fails, the area must be recovered. Standard area recovery procedures apply. DBRC′s GENJCL.CA and GENJCL.RECOV can be used to generate the appropriate recovery jobs. After the recovery of the area, it can be restarted on the sharing IMS subsystems. This will cause the structure or structures to be rebuilt. It might be necessary to change the CFRM policy so that the structures can be built on available coupling facilities.

## 20.10 CF Connection Failures

Sysplex Failure Management (SFM) can be used to cause some connection failures to be treated as if they were structure failures. In these cases, an automatic rebuild on another CF is attempted and the information under 20.7, “Lock Structure Failures” on page 201 or 20.8, “OSAM and VSAM Structure Failures” on page 202 applies.

If there is an active SFM policy with CONNFAIL(YES) specified, REBUILDPERCENT specifications for structure definitions in CFRM policies are used to determine if the rebuild will be attempted. Rebuild will only be attempted if the sum of the weights of the lost connections is equal to or exceeds the specified REBUILDPERCENT. If there is no active SFM policy, the policy does not specify CONNFAIL(YES), and there is no REBUILDPERCENT specified for the structure, or sum of the weights of the lost connections is less than the REBUILDPERCENT, the following will occur.

Since rebuild is not supported for DEDB VSO structures, the REBUILDPERCENT parameter in the CFRM policy should not be specified for them.

## 20.11 CF Connection Failure to Lock Structure

SFM might cause a CF connection failure to be treated as a structure failure. If it does not, the following applies.

The IRLM survives; however, the effects on the IMS subsystems using the IRLM are the same as those for an IRLM abend. Batch jobs are abended, and IMS TM and DBCTL systems do not schedule PSBs or IMS transactions. The procedure will be:

- Repair the CF connection.
- If the CF connection cannot be repaired,
  - either

    Treat this as a failure of the MVS system that cannot be repaired. That is, discontinue use of this system in the sysplex.

  - or

    Rebuild the lock structure on a CF to which all systems have access.

- If the CF connection is repaired, or if the structure is rebuilt, the connection of IMS to the IRLM and IRLM to the structure will be automatic.
- Back out abended batch data-sharing update jobs.
- Restart batch data-sharing jobs.

## 20.12 CF Connection Failure to an OSAM or VSAM Structure

SFM might cause a CF connection failure to be treated as a structure failure. If it does not, the following applies.

The IMS system will issue the message:

```
DFS3384I DATA SHARING STOPPED.
```

All SHARELVL=1, 2, and 3 databases are stopped. This applies to both OSAM and VSAM databases even when only one of the structures is lost. The repair procedures are:

- Repair the CF connection.
- If the CF connection cannot be repaired,
  - either

    Treat this as a failure of the MVS system that cannot be repaired. That is, discontinue use of this system in the sysplex.

  - or

    Rebuild the structure on a CF to which all systems have access.

- If the CF connection is repaired, or if the structure is rebuilt, the connection of IMS to the IRLM will be automatic.
- Back out abended batch data-sharing update jobs.
- Restart batch data-sharing jobs.

## 20.13 CF Connection Failure to a DEDB VSO Structure

When connectivity from an IMS to a DEDB VSO structure fails, the actions depend on the availability of a second structure for the area.

If there is another structure, all IMSs discontinue using the structure with the lost connectivity. The remaining structure is used to support sharing.

If there is not another structure, the area is stopped in each IMS which loses connectivity. Their connections become failed-persistent connections. Sharing of the structure is continued only by the IMSs that continue to have connectivity. The IMSs that no longer have connectivity might have committed updates that have not been written to the structure or DASD. These updates will be applied when connectivity is restored. CIs containing these updates will remained locked until then.

### 20.13.1 Procedure with Connectivity to a Second Structure

If sharing continues with a second structure, immediate action is not required. The use of two structures can be reinstituted after all IMS subsystems close the area. The next time that the area is opened, connections to the two structures will be attempted. If connectivity is available, this will reestablish the use of two structures. It might be necessary to change the CFRM policy so that the structures can be built on coupling facilities with available connectivity.

### 20.13.2 Procedure without Connectivity to a Second Structure

If an IMS subsystem lost connectivity to the only structure, it has a failed persistent connection to this structure. Starting the area will attempt a reconnect. If it is successful, sharing will be reestablished for the subsystem. If the subsystem has unapplied committed updates for the area, they will be applied when the connection is reestablished.

If connectivity to the structure cannot be reestablished, the area must be recovered. This will apply any committed unapplied updates to the area. Standard area recovery procedures can be used. DBRC′s GENJCL.CA and GENJCL.RECOV will generate the appropriate recovery jobs. After the recovery of the area, it can be restarted on the sharing IMS subsystems. This will cause the structure or structures to be rebuilt. It might be necessary to change the CFRM policy so that the structures can be built on available coupling facilities.

## 20.14 Disaster Recovery

The introduction of block level data-sharing into an IMS environment has definite implications on an installation′s disaster recovery procedures. The degree of impact will depend on the existing procedures.

Four different disaster recovery planning scenarios will be described, along with their block level data-sharing implications.

### 20.14.1 Image Copy Only Disaster Recovery

In this scenario, IMS database activity is periodically quiesced and batch image copies (static image copies, not fuzzy online or concurrent image copies) are taken on the databases. The resulting batch image copies are then sent to the disaster recovery site and will be the sole source of database recovery in the case of a disaster situation.

The database quiesce operation consists of either terminating all IMS subsystems or issuing /DBR or /DBD commands for all of the related (either IMS maintained or application maintained) databases on all of the data-sharing systems. The GLOBAL option on the /DBR or /DBD command could be used without major impact since the IMS Batch Image Copy utility ignores the DBRC prohibit authorizations flag (set on by /DBR GLOBAL) and the DBRC read only flag (set on by /DBD GLOBAL).

How often the quiesce operation occurs depends on the amount of data an enterprise can afford to lose in case of a disaster. For example, if the databases are quiesced once a day following the over-night batch cycle (close of business), then up to a day′s worth of activity could be lost.

There are no data-sharing implications for this type of disaster recovery, since the database recovery operations at the disaster recovery site will be using only the batch image copy information and will not be using log or change accumulation information.

### 20.14.2 Time-Stamp Recovery Disaster Recovery

In this scenario, the enterprise establishes periodic time-stamp recovery points for the IMS databases. The logs or the change accumulation data sets resulting from the logs (block level data-sharing generally forces the logs to be processed by the change accumulation utility prior to recovery operations) are then sent to the disaster recovery site, along with the image copies, and are used to recover the databases to the latest time-stamp recovery point in case of a disaster.

In order to establish a time-stamp recovery point, the enterprise has to issue a /DBD or /DBR commands (without the NOFEOV parameter) for all of the related databases (either IMS maintained or application maintained) on all of the sharing IMS subsystems, wait for the /DBR or /DBD commands to complete on all of the sharing IMS subsystems, and then issue /STA commands on all of the sharing IMS subsystems to restore full database availability. It should be noted that a /DBR or /DBD command for a database will fail if a BMP is currently scheduled that is sensitive to the database.

How often the enterprise establishes these time-stamp recovery points is a trade-off between data currency (how much activity can the enterprise afford to lose when a disaster occurs) and data availability (databases cannot be updated while the time-stamp recovery point is being established). In general, one would expect that time-stamp recovery points would be separated by hours. Thus, multiple hours of activity would be lost if a disaster recovery situation were to occur.

There are no data-sharing implications, other than the requirement to perform change accumulation processing, for this type of disaster recovery situation. This is due to the fact that the recovery would be to a valid, time-stamp recovery point where database activity had been quiesced.

### 20.14.3 Latest Archived Log Disaster Recovery

In this scenario, log data (SLDS) is sent to the disaster recovery site when it is archived either as part of the archive process (remote tape or DASD connected through channel extender technology) or shipped following the archive process. A backup copy of the RECONs is created and sent to the disaster recovery site at the same time. In case of a disaster, the RECONs is restored, the databases would be recovered to the end of the latest available log, and then batch backout would be run for any PSBs that were in-flight at the end of the latest available log.

This type of disaster recovery scenario *will not work* in a block level data-sharing environment unless you use a product, such as IMS Recovery Saver (IBM product number 5655-A68). Since multiple IMS subsystems, each producing its own recovery log data stream, are concurrently updating the databases, the database recovery information is contained in multiple log data streams. This poses a number of problems.

- There are multiple *latest archived logs*, one for each sharing IMS subsystem.

- Since there is no absolute coordination between IMS subsystems for OLDS switching, each of the latest archived logs ends at a different time.
- Since there is always missing log data, there is no way that the logs can be merged to produce a change accumulation data set acceptable to the data base recovery utility.

The net result is that the logs and resultant change accumulation data set cannot be used for recovery.

The IMS Recovery Saver provides a solution to these problems by:

1. Analyzing a backup copy of the RECONs to determine which log data streams were active at the time of the RECON backup.
2. Analyzing the log data streams to determine the latest point of log data stream consistency.
3. Copying and truncating the appropriate log data sets to achieve log data stream consistency.
4. Performing all necessary maintenance, including building backout records for in-flight and in-doubt units of recovery, against the RECON data sets so that the databases can be recovered using your standard database recovery utilities of choice.
5. Producing a list of units of recovery (PSBs), by the IMS subsystem, that need to be backed out.

The use of IMS Recovery Saver will allow the user to continue to use a latest archived log disaster recovery methodology and will also assist the disaster recovery process by automating all of the maintenance operations against the backup copy of the RECONs. In addition, IMS Recovery Saver will allow the user to specify the log data stream cut-off or truncation time if recovery to the latest possible state is not desired for business reasons or coordination with other subsystems, such as DB2.

For additional information, see *IMS/ESA Recovery Saver User′s Guide and Reference*, SC26-9191.

### 20.14.4 Real-Time Electronic Log Vaulting Disaster Recovery

In this scenario, the IMS/ESA RSR (Remote Site Recovery) feature is used to transmit full log (OLDS or batch) buffers to the disaster recovery site as they are written to the OLDS or batch log. Since log data is transmitted to the disaster recovery site when it is written to the OLDS or batch log, there will be minimal data lost (partial log buffer plus anything flowing down the line) when a disaster occurs.

Since the IMS/ESA RSR features (both Recovery Level Tracking and Database Level Tracking—*shadowing*) support block-level data-sharing, there are minimal data-sharing implications.

When shadow databases are being maintained, RSR will do the necessary log data stream merging internally. When a disaster recovery take-over situation occurs, RSR will automatically truncate the logs to a consistent state. The only data-sharing implication is the fact that the logs must be processed by change accumulation prior to the use of the IMS Database Recovery utility, but this is not unique to disaster recovery.

# Appendix A. Naming Convention Suggestions

The following suggestions for naming conventions are consistent with those made in *OS/390 Parallel Sysplex Application Migration*, GC28-1863 and used in *DB2 for OS/390 Version 5 DB2 Data Sharing:Planning and Administration*, SC26-8961.

- IMS data sharing group and shared queues group

  The IMS product publications refer to the IRLM GROUP parameter as the data sharing group name. It is used as the XCF group name for the IRLMs. All IRLMs in the data sharing group must specify the same value since they will use the XCF group for communicating with each other. Using this name for other purposes is useful. It can be the identifier for other elements in the data sharing group and shared queues group. These groups can be called an IMSplex.

  An indication of the IMSplex will be needed as part of other names. For example, a subsystem name should indicate in which IMSplex the subsystem participates. Since other names have size restrictions, the IMSplex should also be identified with one character that can be used as part of a name.

  An implementation of these recommendations for a production IMS system could be the following. For a production IMSplex, use IMSP as the IMSplex name. This would also be the IRLM group name. Use *P* as the one character identification for the IMSplex. For shared queues, IMSP would be used for the SQGROUP parameter for IMS and the CQSGROUP parameter for CQS.

- Subsystem names and IDs

  Subsystem names are limited to four characters, and are created using the following conventions:

  **ctgi** where:

  **c** collection

  This identifies a collection of logically related subsystems. All IMS and IRLM subsystems in a data sharing group or IMSplex would be in the same collection. Similarly, all of the IMSs sharing a set of queues would be in the same collection. One collection would typically contain a data sharing group and a shared queues group. The collection would also include any CICS regions using the IMSs for DBCTL services. A collection would include one IMS data sharing group and one group of IMS systems sharing queues.

  **t** type of subsystem or region

  Values could be:

  - D for IMS
  - Q for CQS
  - F for FDBR
  - I for IRLM used by IMS
  - B for DB2

- J for IRLM used by DB2
- A for CICS Application Owning Region
- T for CICS Terminal Owning Region
- C for a CICSPlex SM CMAS

**g** recovery group

This identifies a set of subsystems or regions that must reside in the same MVS. It could identify the MVS system on which they normally execute. For example, it could identify an IMS DBCTL subsystem and the CICS AORs that use it.

**i** instance or iteration

This identifies one of multiple subsystems on the same collection, type, and recovery group. For example, it could could be one of multiple CICS AORs that are in the same collection, type, and recovery group.

If the collection identifier were P for production, the MVS systems were 1, 2, and 3, and there was one IMS and one IRLM per MVS, the subsystem names would be:

IMS:

- PD11
- PD21
- PD31

IRLM:

- PI11
- PI21
- PI31

If an installation chooses to have all IRLMs in a data-sharing group using the same IRLM name, they might substitute a 0 for the *g* (recovery group or MVS) in the name. Our example would then have the following name for all of the IRLMs.

IRLM:

- PI01

- Job names or STC names for IMS regions

  An installation will probably want unique names for their control region, DBRC address space, DLI SAS, dependent regions, CQS, and IRLM procedures. If so, they can use the subsystem name as a prefix in the procedure name. Different procedures for the same subsystem would be identified by the rest of the name. The following could be used:

  **ctgibbbb** where:

  **ctgi** IMSID

  **bbbb**

  - IMS for IMS control region
  - DBRC for DBRC address space
  - DLS for DLI separate address space

    - IRLM for IRLM used by IMS
    - CQS for CQS address space

  For example, the following procedures would be used for IMS subsystem PD11, IRLM PI11, and CQS subsystem PQ11.

  - PD11IMS
  - PD11DBRC
  - PD11DLS
  - PI11IRLM
  - PQ11CQS

  A slight alternative to this naming scheme is the use of the IMS subsystem name in all of the procedures associated with it. For example, the following procedures would be used for IMS subsystem PD11, IRLM PI11, and CQS subsystem PQ11.

  - PD11IMS
  - PD11DBRC
  - PD11DLS
  - PD11IRLM
  - PD11CQS

- VTAM APPLIDs and Generic Resource Group Names

  Each IMS TM system in an IMSplex will require a unique VTAM APPLID. Users of generic resources will need to specify a generic resource name for the IMSplex. For ease of migration purposes, installations might want to use the APPLID of their current IMS TM system as the generic resource group name. This will eliminate the need to change users′ procedures for logons. An APPLID should include an identification of the IMS subsystem that uses it.

  An example of possible names for an installation whose current IMS APPLID is IMSPROD and which will implement a three-way IMSplex is:

  - Generic resource group name - IMSPROD
  - APPLIDs - PD11PROD, PD12PROD, and PD13PROD

  Users of APPC/IMS will also need APPLIDs and generic resource names for APPC/IMS use. They can be developed as the IMS names were developed. The generic resource group name would be the current APPLID, and the new APPLIDs would contain an identification of the IMS subsystems.

  An example of possible names for an installation whose current APPC/IMS APPLID is IMSPAPPC and which will implement a three-way IMSplex is:

  - Generic resource group name - IMSPAPPC
  - APPLIDs - PD11APPC, PD12APPC, and PD13APPC

- Structure names

  Many users of the Coupling Facility either require a specific name for their Coupling Facility structures or limit which names can be used. There is not a uniform naming convention implemented for all of these users. Any structure names that IBM requires for its products will begin either with A-I or with SYS. Installations might want to use a naming convention for IMS and IRLM structures that does does not begin with these letters.

  DB2 requires that all of the structures that it and its IRLM use begin with *groupname_* where groupname is the DB2 group name. IMS users can implement a similar scheme by using the IMS data sharing group name in the structure names. The data sharing group name is also the IMSplex name. If this is followed by an identification of the type of structure, the name would be unique.

Structure names are limited to 16 characters.

The following scheme could be used:

**groupname_xxxxxxxx** where:

**groupname** IMSplex name

If DEDB VSO structures are used, the groupname would have to be six characters or less because the structure name is limited to 16 characters and this scheme has an ″_′ and up to nine other characters in the rest of the name for DEDB VSO structures.

**XXXXXXXX**

- ″IMSIRLM″ for the lock structure
- ″IMSVSAM″ for the VSAM buffer invalidation structure
- ″IMSOSAM″ for the OSAM buffer invalidation structure
- yyyyyyyyn for DEDB VSO structures where

  **yyyyyyyy** Area name

  **n** ″1″ or ″2″ when two structures are used
- ″IMSMSGP″ for the primary message queues structure
- ″IMSMSGO″ for the overflow message queues structure
- ″IMSEMHP″ for the primary EMH queues structure
- ″IMSEMHO″ for the overflow EMH queues structure
- ″IMSLOGM″ for the logger structure for the message queue log stream
- ″IMSLOGE″ for the logger structure for the EMH queue log stream

For an IMSplex name of IMSP and two shared DEDB VSO areas with names DA000AAA and DA000BBB, each with dual structures, we would have the following structure names:

- IMSP_IMSIRLM
- IMSP_IMSVSAM
- IMSP_IMSOSAM
- IMSP_DA000AAA1
- IMSP_DA000AAA2
- IMSP_DA000BBB1
- IMSP_DA000BBB2
- IMSP_IMSMSGP
- IMSP_IMSMSGO
- IMSP_IMSEMHP
- IMSP_IMSEMHO
- IMSP_IMSLOGM
- IMSP_IMSLOGE

- IMS system data sets

  For data sets that are shared among IMS systems in an IMSplex, an indicator of the IMSplex should be included as a qualifier in the data set name. This would correspond to the collection in the subsystem naming convention. For example, if the IMSplex name were IMSP, the name of the ACBLIB could be:

  - IMS.IMSP.ACBLIB

  For data sets that are unique to an IMS system, the IMSID can be used as a qualifier in the data set name. For example, if the IMSID is PD11, the name of the RDS could be:

  - IMS.IMSP.PD11.RDS

- Database data sets

  It is unlikely that the names of database data sets for shared databases will need to be changed. Additional database data sets are not required for data sharing. If multiple data sharing groups are used, it might be advisable to include the IMSplex name as a qualifier in the data set names.

- Application data sets

  Application data sets are non-data base data sets that are used as part of applications. Batch program input and output files are examples of these. If the the data sets must be cloned, because a data set is needed for each IMS subsystem, the IMS subsystem name should be used as a qualifier in the name. If cloning is not required, the data set name does not need to be changed.

# Appendix B. IMS System Data Sets

The following tables assume that the target IMS systems are in a CLONED configuration. They define the data sets as one of the following:

- Must Be Unique
- Must Be Shared
- Probably Unique
- Probably Shared

## B.1 Data Sets That Must Be Unique

The following IMS system data sets *must be unique* among the IMS subsystems within the data-sharing group.

| DATA SET | CATEGORY | COMMENTS |
|---|---|---|
| IMS Log Data Sets | EXECUTE | OLDS, WADS, SLDS, and RLDS |
| Message Queue Data Sets | EXECUTE | QBLKS, SHMSG, SHMSG1-9, LGMSG, LGMSG1-9, QBLKSL, SHMSGL, and LGMSGL.<br>The ″L″ suffixes are for XRF systems only. |
| Restart Data Sets | EXECUTE | RDS and RDS2 (XRF) |
| Modify Status Data Set | EXECUTE | MODSTAT and MODSTAT2 (XRF) |
| IMS Monitor Data Set | EXECUTE | IMSMON |
| External Trace Data Set | EXECUTE | DFSTRA01, DFSTRA02, and DFSTRA0T. |
| MSDB Data Sets | EXECUTE | MSDBCP1-2, MSDBCP3-4 (XRF only), MSDBDUMP, and MSDBINIT |
| IMS TM SYSOUT Data Sets | EXECUTE | These are data sets defined in IMS system definitions with the UNITYPE parameter of the LINEGRP macro specified as either DISK, PUNCH, READER, PRINTER, TAPE, or SPOOL. |
| JCLOUT | EXECUTE | These data sets hold the output of the GENJCL command and are included in the DBRC address space JCL. |
| CQS Checkpoint Data Sets | EXECUTE | A data set is required for each shared queues structure pair. One is used for full-function, and one is used for EMHQ. |

## B.2 Data Sets That Must Be Shared

The following IMS system data sets *must be shared* by all IMS subsystems within the data-sharing group.

| DATA SET | CATEGORY | COMMENTS |
|---|---|---|
| RECONs | EXECUTE | RECON1-3. |
| CQS Structure Recovery Data Sets | EXECUTE | A pair of data sets is required for each shared queue structure pair. One is used for full-function, and one is used for EMHQ. |
| MVS Logger Spill Data Sets | EXECUTE | Data sets are required for each shared queues structure pair. Logger spill data sets are associated with MVS log streams. |

## B.3 Data Sets That Are Probably Unique

The following IMS system data sets *can be shared* by all IMS subsystems within the data-sharing group, but are *probably unique*.

| DATA SET | CATEGORY | COMMENTS |
|---|---|---|
| DFSTCF | EXECUTE | This is the IMS TM TCO Script Library. The initial script must be in member DFSTCF. |

## B.4 Data Sets That Are Probably Shared

The following IMS system data sets *are probably shared* by all IMS subsystems within the data-sharing group.

| DATA SET | CATEGORY | COMMENTS |
|---|---|---|
| MFS Referral Library | CUSTOM | This data set is used by the MFS Language Utility. |
| LGENIN and LGENOUT | CUSTOM | These data sets are used during the system definition process when the Large System Generation (LGEN) option is specified. |
| OBJDSET | CUSTOM | These data sets are used during the system definition process when the Large System Generation (LGEN) option is specified. |
| OPTIONS | CUSTOM | These data sets are used during the system definition process when the Large System Generation (LGEN) option is specified. |
| ACBLIBs | EXECUTE | ACBLIB is the staging library. ACBLIBA and ACBLIBB are read by executing IMS systems. It is required that DMBs for a shared database be the same. For cloned systems, the PSBs used by each system should be the same. For these reasons, it is assumed that one instance of each of these data sets will be used by all cloned IMS systems. |
| FMTLIBs | EXECUTE | FORMAT is the staging library. FORMATA, FORMATB, and TFORMAT are read by executing IMS systems. |

| DATA SET | CATEGORY | COMMENTS |
|---|---|---|
| JCLPDS | EXECUTE | This is the source of skeletal JCL. The skeletal JCL for GENJCL.ARCHIVE includes a STEPLIB pointing to a RESLIB. It is unlikely that this is a problem, since it is only used to find a library with the ARCHIVE program (DFSUARC0). Also, the data sets have DSNs that include .%SSID. qualifiers. |
| IMS RESLIB | EXECUTE | If different IMS systems are to be run at different release levels, they will require unique RESLIBs. |
| MODBLKSx | EXECUTE | These data sets are associated with a RESLIB. If unique RESLIBs are used, there will probably be unique versions of these data sets. |
| MATRIXx | EXECUTE | These data sets are associated with a RESLIB. If unique RESLIBs are used, there will probably be unique versions of these data sets. |

# Appendix C. Sample USERVAR Exit for Network Balancing

*This appendix applies only to IMS TM.*

The following sample exit is provided as an example of a USERVAR exit that could be used to implement workload balancing through the network balancing approach (see 9.2, “Network Workload Balancing” on page 66). It is usable with both IMS/ESA 5.1 and IMS/ESA 6.1. It is supplied primarily for IMS/ESA 5.1 users who do not have support for VTAM generic resources.

This sample exit has not been submitted to any formal IBM test and is distributed on an *As Is* basis without any warranty either expressed or implied. The use of this sample exit depends on the customer′s ability to evaluate and integrate it into the customer′s operational environment.

The sample will simulate generic logon support for IMS by distributing logon requests by hashing or randomizing the node name requesting logon. Since a repeatable hashing technique is used (the same LU name always hashes to the same value), successive logon requests from the same node will be directed to the same IMS system (APPLID).

The exit routine is table driven (the table is assembled with the exit) and can handle translation requests for up to five different USERVARs. The ability to handle multiple USERVARs is critical since there is only one VTAM USERVAR exit within a VTAM system. Logon requests for each USERVAR managed by the exit can be distributed among up to ten different IMS system (APPLIDs).

The limits on the number of USERVARs managed and the number of APPLIDs per USERVAR are easily changed.

The sample exit also allows the logon distribution characteristics to be changed through operator command for one of the managed USERVARs.

For information regarding the VTAM USERVAR exit, ISTEXCUV, see the VTAM Customization manual for your VTAM release. For information regarding the VTAM definition of USERVARs and a description of the VTAM commands used to specify the use of, activation and replacement of, and how to pass parameter information to a USERVAR exit, see the VTAM Operation manual for your VTAM release.

```
UVAR     TITLE ′SAMPLE USERVAR EXIT TO DISTRIBUTE LOGON REQUESTS TO MULX00010000
               TIPLE IMS SYSTEMS′                                       00020000
ISTEXCUV CSECT ,                                                        00030000
ISTEXCUV AMODE 31                                                       00040000
ISTEXCUV RMODE ANY                                                      00050000
         PRINT NOGEN                                                    00060000
         SPACE 1                                                        00070000
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * 00080000
*                                                                     * 00090000
*  MODULE NAME:  ISTEXCUV                                             * 00100000
*                                                                     * 00110000
*  ATTRIBUTES:  AMODE 31, RMODE ANY, REENTRANT                        * 00120000
*                                                                     * 00130000
*  PURPOSE:  THIS EXIT WILL SIMULATE GENERIC LOGON SUPPORT FOR IMS BY * 00140000
*            DISTRIBUTING LOGON REQUESTS BASED ON HASHING THE NODE    * 00150000
*            NAME REQUESTING LOGON.  SINCE A REPEATABLE HASHING       * 00160000
*            TECHNIQUE IS USED, SUCCESSIVE LOGON REQUESTS REQUESTS    * 00170000
*            FROM THE SAME LU WILL BE DIRECTED TO THE SAME IMS SYSTEM * 00180000
*            (ASSUMING THAT THIS EXIT HAS NOT BEEN UPDATED WITH NEW   * 00190000
*            DISTRIBUTION PARAMETERS).                                * 00200000
*                                                                     * 00210000
*            THE EXIT IS TABLE DRIVEN AND CAN HANDLE REQUESTS FOR UP  * 00220000
*            TO 5 DIFFERENT USERVARS.  FOR EACH USERVAR THAT IS       * 00230000
*            CONTROLLED, LOGON REQUESTS CAN BE DISTRIBUTED AMONG UP   * 00240000
*            TO 10 DIFFERENT IMS SYSTEMS (APPLIDS).                   * 00250000
*                                                                     * 00260000
*            THIS SAMPLE EXIT HAS NOT BEEN SUBMITTED TO ANY FORMAL    * 00270000
*            IBM TEST AND IS DISTRIBUTED ON AN ’AS IS’ BASIS WITHOUT  * 00280000
*            ANY WARRANTY EITHER EXPRESSED OR IMPLIED.  THE USE OR    * 00290000
*            IMPLEMENTATION OF THIS SAMPLE EXIT IS A CUSTOMER         * 00300000
*            RESPONSIBILITY AND DEPENDS ON THE CUSTOMER’S ABILITY TO  * 00310000
*            EVALUATE AND INTEGRATE IT INTO THE CUSTOMER’S            * 00320000
*            OPERATIONAL ENVIRONMENT.                                 * 00330000
*                                                                     * 00340000
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * 00350000
         TITLE ’SAMPLE USERVAR EXIT TO DISTRIBUTE LOGON REQUESTS TO MULX00360000
               TIPLE IMS SYSTEMS - INITIALIZATION’                      00370000
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * 00380000
*                                                                     * 00390000
*  DEFINE REGISTER EQUATES                                            * 00400000
*  SAVE REGISTERS                                                     * 00410000
*  ESTABLISH EXIT BASE                                                * 00420000
*  DETERMINE TYPE OF CALL AND BRANCH TO APPROPRIATE ROUTINE           * 00430000
*  ISSUE WTO IF TYPE OF CALL UNKNOWN AND RETURN                       * 00440000
*                                                                     * 00450000
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * 00460000
         SPACE 1                                                        00470000
R0       EQU   0                                                        00480000
R1       EQU   1                                                        00490000
R2       EQU   2                                                        00500000
R3       EQU   3                                                        00510000
R4       EQU   4                                                        00520000
R5       EQU   5                                                        00530000
R6       EQU   6                                                        00540000
R7       EQU   7                                                        00550000
R8       EQU   8                                                        00560000
R9       EQU   9                                                        00570000
R10      EQU   10                                                       00580000
R11      EQU   11                                                       00590000
R12      EQU   12                                                       00600000
R13      EQU   13                                                       00610000
R14      EQU   14                                                       00620000
R15      EQU   15                                                       00630000
         SPACE 1                                                        00640000
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * 00650000
*                                                                     * 00660000
*  THE EYECATCHER IN THE SAVE MACRO ’V001’ SHOULD BE CHANGED WHEN THE * 00670000
*  VERSION OF THIS EXIT IS CHANGED.  THIS WILL ALLOW YOU TO           * 00680000
*  CORRELATE A DUMP WITH AN EXIT ASSEMBLY.                            * 00690000
*                                                                     * 00700000
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * 00710000
         SPACE 1                                                        00720000
         SAVE  (14,12),,&SYSDATE..V001                                  00730000
         USING ISTEXCUV,R12       DEFINE EXIT BASE                      00740000
         LR    R12,R15            LOAD EXIT BASE                        00750000
         CH    R0,HEX04           IS ENTRY FOR EXIT INVOCATION?         00760000
         BE    INVOKE             YES, BRANCH TO ROUTINE                00770000
         CH    R0,HEX10           IS ENTRY FOR EXIT ACTIVATION?         00780000
         BE    ACT                YES, BRANCH TO ROUTINE                00790000
         CH    R0,HEX18           IS ENTRY FOR EXIT REPLACEMENT?        00800000
         BE    REPL               YES, BRANCH TO ROUTINE                00810000
         CH    R0,HEX20           IS ENTRY FOR EXIT DEACTIVATION?       00820000
         BE    DEACT              YES, BRANCH TO ROUTINE                00830000
         CH    R0,HEX40           IS ENTRY FOR VTAM TERMINATION?        00840000
         BE    INITRET             YES, BRANCH TO ROUTINE               00850000
         SPACE 1                                                        00860000
         WTO   ’REGISTER 0 CONTENTS TO USERVAR EXIT UNEXPECTED’,       X00870000
               ROUTCDE=(10),       SYSTEM ERROR INFORMATION            X00880000
               DESC=(12),          IMPORTANT INFORMATION MESSAGE       X00890000
               LINKAGE=BRANCH      USE BRANCH ENTRY                     00900000
         SPACE 1                                                        00910000
INITRET  DS    0H                                                       00920000
         RETURN (14,12),,RC=0      RETURN                               00930000
         TITLE ’SAMPLE USERVAR EXIT TO DISTRIBUTE LOGON REQUESTS TO MULX00940000
               TIPLE IMS SYSTEMS - EXIT INVOCATION’                     00950000
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * 00960000
*                                                                     * 00970000
*  IF NON-TRANSLATION REQUEST THEN                                    * 00980000
*    RETURN                                                           * 00990000
*  IF TRANSLATION REQUEST FOR USERVAR NOT MANAGED BY EXIT THEN        * 01000000
*    SET NOT-TRANSLATED                                               * 01010000
*    RETURN                                                           * 01020000
*  HASH THE OLU NAME GIVING RESULT 1 - 100                            * 01030000
*  SEARCH DIST TABLE FOR CUMULATIVE PERCENT GREATER THAN OR EQUAL    * 01040000
*    TO HASH VALUE                                                    * 01050000
*  SET TRANSLATED NAME TO DIST TABLE ENTRY                            * 01060000
*  SET TRANSLATED                                                     * 01070000
*  RETURN                                                             * 01080000
*                                                                     * 01090000
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * 01100000
         SPACE 1                                                        01110000
INVOKE   DS    0H                  EXIT INVOCATION ROUTINE              01120000
         LR    R2,R1               GET PARM ADDR                        01130000
         USING PARM2,R2            SET PARAMETER BASE                   01140000
         L     R3,INVOKEA          GET ADDR OF INVOKE FLAGS             01150000
         CLI   0(R3),X’04’         IS THIS A TRANSLATION REQUEST?       01160000
         BNE   INVOKRET            NO, RETURN                           01170000
         L     R3,USRVPRMS         GET ADDR OF USERVAR PARMS            01180000
         USING USRVPARM,R3         SET USERVAR PARMS BASE               01190000
         L     R4,USERADD1         GET ADDR OF USER FIELD               01200000
         USING USERFLD,R4          SET BASE FOR USER FIELD              01210000
         L     R4,STORADDR         GET ADDR OF GETMAINED STORAGE        01220000
         DROP  R4                  DROP USER FIELD BASE                 01230000
         LA    R4,WORKLEN(,R4)     GET ADDR OF USERVAR TBL              01240000
         USING USRVDSCT,R4         SET BASE FOR USERVAR ENTRY           01250000
INVOKE1  DS    0H                  START OF SEARCH LOOP                 01260000
         CLC   USRVNAME,GENERIC    USERVAR NAME FOUND IN TABLE?         01270000
         BE    INVOKE2             YES, PROCESS                         01280000
         L     R4,USRVNEXT         GET ADDR OF NEXT USERVAR ENTRY       01290000
         LTR   R4,R4               IS THERE A NEXT USERVAR ENTRY?       01300000
         BNZ   INVOKE1             YES, LOOP                            01310000
         B     INVOKRET            NO, USERVAR NOT FOUND, RETURN        01320000
INVOKE2  DS    0H                  USERVAR FOUND, NOW HASH OLU NAME     01330000
         SPACE 1                                                        01340000
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * 01350000
*  VERY SIMPLE HASHING TECHNIQUE OF LOGICALLY ADDING THE FIRST 4      * 01360000
*  BYTES OF THE OLU TO THE LAST 4 BYTES OF THE OLU, DIVIDING BY 100   * 01370000
*  AND ADDING 1 TO THE REMAINDER.                                     * 01380000
*                                                                     * 01390000
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * 01400000
         SPACE 1                                                        01410000
         L     R7,OLUNAME          GET FIRST 4 BYTES OF OLU             01420000
         AL    R7,OLUNAME+4        ADD LAST 4 BYTES OF OLU              01430000
         N     R7,HIOFF            CLEAR HIGH BIT                       01440000
         XR    R6,R6               CLEAR R6                             01450000
         D     R6,=F’100’          DIVIDE BY 100                        01460000
         LA    R6,1(,R6)           ADD 1 TO REMAINDER                   01470000
         SPACE 1                                                        01480000
         L     R5,USRVDIST         GET ADDR OF DISTRIBUTION TABLE       01490000
         USING DISTDSCT,R5         SET DISTRIBUTION TABLE ENTRY BASE    01500000
INVOKE3  DS    0H                  START OF DIST TBL SEARCH LOOP        01510000
         C     R6,DISTHIGH         COMPARE HASH VALUE TO CUMM PERCENT   01520000
         BNH   INVOKE4             BRANCH IF LESS THAN OR EQUAL         01530000
         LA    R5,DISTLEN(,R5)     POINT TO NEXT ENTRY                  01540000
         B     INVOKE3             LOOP                                 01550000
INVOKE4 DS     0H                  CORRECT DISTRIBUTION ENTRY FOUND     01560000
         MVC   TRANVAL,DISTAPPL    MOVE APPLID TO PARM LIST             01570000
         OI    FLAGS,TRANSLAT      SET TRANSLATE FLAG ON                01580000
INVOKRET DS    0H                  RETURN                               01590000
         RETURN (14,12),,RC=0                                           01600000
         DROP  R2                  DROP PARAMETER BASE                  01610000
         DROP  R3                  DROP USERVAR PARMS BASE              01620000
         DROP  R4                  DROP USERVAR TABLE ENTRY BASE        01630000
         DROP  R5                  DROP DISTRIBUTION TABLE ENTRY BASE   01640000
         TITLE ’SAMPLE USERVAR EXIT TO DISTRIBUTE LOGON REQUESTS TO MULX01650000
               TIPLE IMS SYSTEMS - EXIT ACT/REPL’                       01660000
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * 01670000
*                                                                     * 01680000
*  OBTAIN STORAGE FOR USERVAR LIST, DISTRIBUTION LIST, AND USER DATA  * 01690000
*    WORKING STORAGE                                                  * 01700000
*  IF STORAGE REQUEST FAILED THEN                                     * 01710000
*    ISSUE ERROR WTO AND RETURN                                       * 01720000
*  BUILD USERVAR AND DISTRIBUTION LISTS IN OBTAINED STORAGE           * 01730000
*  IF USER DATA PROVIDED THEN                                         * 01740000
*    VALIDATE USERVAR NAME AND DISTRIBUTION PARAMETERS                * 01750000
*    IF PARAMETER ERROR THEN                                          * 01760000
*      ISSUE ERROR WTO AND RETURN                                     * 01770000
*    ELSE                                                             * 01780000
*      OVER-RIDE SPECIFIED USERVAR PARAMETER DATA                     * 01790000
*  RETURN                                                             * 01800000
*                                                                     * 01810000
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * 01820000
         SPACE 1                                                        01830000
ACT      DS    0H                                                       01840000
REPL     DS    0H                                                       01850000
         LR    R7,R1               SAVE PARAMETER ADDR                  01860000
         L     R2,WRKSPCLN         GET SIZE OF REQUIRED STORAGE         01870000
         STORAGE OBTAIN,COND=YES,  OBTAIN STORAGE FOR LISTS            X01880000
               LENGTH=(2),ADDR=(3)                                      01890000
         LTR   R15,R15             WAS STORAGE REQUEST SUCCESSFUL?      01900000
         BZ    ACT1                YES, BRANCH OVER ERROR MESSAGE       01910000
         SPACE 1                                                        01920000
         WTO   ’STORAGE OBTAIN FAILED IN USERVAR EXIT’,                X01930000
               ROUTCDE=(10),       SYSTEM ERROR INFORMATION            X01940000
               DESC=(12),          IMPORTANT INFORMATION MESSAGE       X01950000
               LINKAGE=BRANCH      USE BRANCH ENTRY                     01960000
         B     ACTRET              RETURN                               01970000
         SPACE 1                                                        01980000
ACT1     DS    0H                  RELOCATE TABLE ENTRIES               01990000
         LA    R4,USRVTBL1         GET ADDR OF 1ST USERVAR ENTRY DEF    02000000
         LA    R5,WORKLEN(,R3)     GET ADDR OF USERVAR TBL              02010000
         USING USRVDSCT,R5         DEFINE USERVAR ENTRY BASE            02020000
ACT2     DS    0H                  START OF RELOCATION LOOP             02030000
         MVC   0(USRVLEN+DISTLEN*DISTMAX,R5),0(R4) MOVE ENTIRE ENTRY    02040000
         LA    R6,USRVLEN(,R5)     GET ADDR OF DISTRIBUTION LIST        02050000
         ST    R6,USRVDIST         STORE IN USERVAR ENTRY               02060000
         L     R6,USRVNEXT         GET ADDR OF NEXT USERVAR ENTRY       02070000
         LTR   R6,R6               IS THERE ANOTHER?                    02080000
         BZ    ACT3                NO, GO CHAIN USERVAR ENTRIES         02090000
         LA    R4,USRVLEN+DISTLEN*DISTMAX(,R4) POINT TO NEXT USERVAR    02100000
         LA    R5,USRVLEN+DISTLEN*DISTMAX(,R5) POINT TO NEXT SPACE      02110000
         B     ACT2                GO RELOCATE NEXT ENTRY               02120000
         DROP  R5                  DROP USERVAR ENTRY BASE              02130000
         SPACE 1                                                        02140000
ACT3     DS    0H                                                       02150000
```

```
         LA    R5,WORKLEN(,R3)     GET ADDR OF USERVAR TBL             02160000
ACT4     DS    0H                  START OF USERVAR ENTRY CHAINING LOOP 02170000
         USING USRVDSCT,R5         SET USERVAR BASE TO RELOCATED ENTRY  02180000
         L     R6,USRVNEXT         GET ADDR OF NEXT ENTRY              02190000
         LTR   R6,R6               IS THERE A NEXT ENTRY?              02200000
         BZ    ACT5                NO,                                 02210000
         LA    R6,USRVLEN+DISTLEN*DISTMAX(,R5) GET ADDR OF NXT REL ENT  02220000
         ST    R6,USRVNEXT         STORE IN RELOCATED ENTRY            02230000
         DROP  R5                  DROP USERVAR BASE                   02240000
         LR    R5,R6               GET ADDR OF NEXT ENTRY              02250000
         B     ACT4                PROCESS NEXT ENTRY                  02260000
ACT5     DS    0H                  STORE ADDR OF OBT STOR IN USER FIELD 02270000
         USING PARM1,R7            DEF BASE FOR INPUT PARM LIST        02280000
         L     R2,USERADDR         GET ADDR OF USER FIELD              02290000
         L     R8,PARMADDR         GET ADDR OF USER DATA               02300000
         DROP  R7                  DROP INPUT PARM LIST BASE           02310000
         USING USERFLD,R2          DEF BASE FOR USER FIELD             02320000
         ST    R3,STORADDR         SAVE OBTAINED STORAGE ADDR          02330000
         MVC   VERID,VERSION       SAVE EXIT VERSION ID                02340000
         DROP  R2                  DROP INPUT PARM LIST BASE           02350000
         SPACE 1                                                       02360000
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * 02370000
*                                                                     * 02380000
*  THIS SECTION PROCESSES USER DATA PASSED AS PART OF THE ACTIVATE OR * 02390000
*  REPLACE COMMAND                                                    * 02400000
*                                                                     * 02410000
*  THE USER DATA MUST BE IN THE FORMAT OF                             * 02420000
*                                                                     * 02430000
*    USER-VAR,XXX,XXX,XXX,XXX,XXX,XXX,XXX,XXX,XXX,XXX                 * 02440000
*                                                                     * 02450000
*    WHERE USER-VAR IS THE 8 CHARACTER (PAD WITH BLANKS) USER NAME    * 02460000
*    TO BE OVER-RIDEN                                                 * 02470000
*                                                                     * 02480000
*    XXX IS A 3 DIGIT (MUST BE 3 DIGITS) DISTRIBUTION PERCENTAGE      * 02490000
*    TO BE OVER-RIDDEN                                                * 02500000
*                                                                     * 02510000
*    AT LEAST ONE OVER-RIDE PERCENTAGE MUST BE SPECIFIED AND UP TO    * 02520000
*    10 CAN BE SPECIFIED.  OVER-RIDE PERCENTAGE DISTRIBUTION ENTIRES  * 02530000
*    APPLY TO THE CORRESPONDING ENTRIES IN THE APPROPRIATE USERVAR    * 02540000
*    DISTRIBUTION TABLE COMPILED IN THIS EXIT ROUTINE.                * 02550000
*                                                                     * 02560000
*    NEW ENTRIES IN THE DISTRIBUTION TABLE CANNOT BE ADDED VIA USER   * 02570000
*    DATA.  IN ADDITION, THE MERGING OF THE OVER-RIDE AND COMPILED    * 02580000
*    DISTRIBUTION ENTRIES MUST RESULT IN A TABLE WHERE EACH           * 02590000
*    DISTRIBUTION ENTRY IS GREATER THAN OR EQUAL TO THE PRIOR AND AT  * 02600000
*    LEAST ONE DISTRIBUTION PERCENTAGE IS 100.                        * 02610000
*                                                                     * 02620000
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * 02630000
         SPACE 1                                                       02640000
         LTR   R8,R8               IS ADDR OF USER DATA ZERO?          02650000
         BZ    ACTRET              YES, RETURN                         02660000
         LH    R2,0(,R8)           GET USER DATA LENGTH                02670000
         CH    R2,=H'0'            IS USER DATA LENGTH ZERO?           02680000
         BE    ACTRET              YES, RETURN                         02690000
         CH    R2,=H'12'           IS USER DATA LENGTH AT LEAST 12?    02700000
         BL    PARMERR             NO, INVALID FORMAT                  02710000
         CH    R2,=H'48'           IS USER DATA LENGTH GREATER THAN 48? 02720000
         BH    PARMERR             YES, INVALID FORMAT                 02730000
         TM    1(R8),X'03'         IS USER DATA LENGTH A MULT OF 4?    02740000
         BNZ   PARMERR             NO, INVALID FORMAT                  02750000
         LA    R4,2(,R8)           SET R4 TO USER DATA                 02760000
         LA    R5,0(R2,R4)         SET R5 TO END OF USER DATA + 1      02770000
*  R3 POINTS TO GETMAINED STORAGE, R4 POINTS TO USER DATA, AND R5      02780000
*  POINTS TO BYTE AFTER USER DATA                                      02790000
         USING WORKSTOR,R3         DEFINE WORKING STORAGE BASE         02800000
         MVI   WSSWITCH,X'00'      CLEAR WORKING STORAGE SWITCH        02810000
         LA    R7,DISTMAX          SET LOOP COUNTER                    02820000
         LA    R6,WSDIST           POINT FOR FIRST WS DIST ENTRY       02830000
ACT6     DS    0H                  INIT WS LOOP                        02840000
         MVC   0(L'WSDIST,R6),=F'-1' SET WS DIST ENTRY TO -1           02850000
         LA    R6,L'WSDIST(,R6)    POINT TO NEXT WS DIST ENTRY         02860000
         BCT   R7,ACT6             LOOP DISTMAX TIMES                  02870000
*  SAVE USERVAR NAME FROM USER DATA AND CHECK DIST SYNTAX              02880000
         MVC   WSUSRVAR,0(R4)      MOVE USERVAR NAME TO WS             02890000
         LA    R6,8(,R4)           GET ADDR OF FIRST OVER-RIDE         02900000
         LA    R7,WSDIST           GET ADDR OF FIRST WS DIST ENTRY     02910000
ACT7     DS    0H                  SYNTAX CHECKING LOOP                02920000
         CLI   0(R6),C','          IS 1ST CHAR A COMMA?                02930000
         BNE   PARMERR             NO, ERROR                           02940000
         TRT   1(3,R6),TRTABLE     ARE NEXT 3 CHARS NUMERIC?           02950000
         BNZ   PARMERR             NO, ERROR                           02960000
         PACK  WSDOUBLE,1(3,R6)    PACK DIST OVER-RIDE                 02970000
         CVB   R2,WSDOUBLE         CONVERT DIST OVER-RIDE TO BINARY    02980000
         ST    R2,0(R7)            STORE IN WS DIST ENTRY              02990000
         LA    R6,4(,R6)           GET ADDR OF NEXT OVER-RIDE          03000000
         LA    R7,L'WSDIST(,R7)    GET ADDR OF NEXT WS DIST ENTRY      03010000
         CR    R6,R5               POINTING BEYOND USER DATA?          03020000
         BL    ACT7                NO, LOOP                            03030000
*  CHECK IF USERVAR SPECIFIED IS CONTROLLED                            03040000
         LA    R11,WORKLEN(,R3)    GET ADDR OF 1ST USERVAR             03050000
         USING USRVDSCT,R11        DEFINE USERVAR ENTRY BASE           03060000
ACT8     DS    0H                  SEARCH USERVAR TABLE LOOP           03070000
         CLC   USRVNAME,WSUSRVAR   SPECIFIED USERVAR NAME FOUND?       03080000
         BE    ACT9                YES, EXIT LOOP                      03090000
         L     R11,USRVNEXT        NO, GET ADDR OF NEXT USERVAR ENTRY  03100000
         LTR   R11,R11             IS IT ZERO?                         03110000
         BNZ   ACT8                NO, LOOP                            03120000
         B     PARMERR             YES, ERROR                          03130000
*  CHECK WS DIST ENTRIES FOR INCREASING VALUES AND VALUES BETWEEN      03140000
*  0 AND 100                                                           03150000
ACT9     DS    0H                                                      03160000
         LA    R6,WSDIST           GET ADDR OF FIRST DIST ENTRY        03170000
         XC    WSOLDHI,WSOLDHI     CLEAR CURRENT HIGH ENTRY HOLDER     03180000
         LA    R7,DISTMAX          SET LOOP COUNTER                    03190000
ACT10    DS    0H                  CHECK DIST VALUE LOOP               03200000
         L     R8,0(,R6)           GET WS DISTRIBUTION ENTRY           03210000
         LTR   R8,R8               IS IT NEGATIVE?                     03220000
         BM    ACT11               YES, GET OUT OF LOOP                03230000
         C     R8,=F'100'          IS IT GREATER THAN 100?             03240000
         BH    PARMERR             YES, ERROR                          03250000
         CH    R8,WSOLDHI          IS IT LESS THAN PRIOR HIGH VALUE?   03260000
         BL    PARMERR             YES, ERROR                          03270000
         STH   R8,WSOLDHI          SAVE CURRENT VALUE AS NEW HIGH      03280000
         LA    R6,L'WSDIST(,R6)    BUMP TO NEXT ENTRY                  03290000
         BCT   R7,ACT10            LOOP DISTMAX TIMES                  03300000
*  CHECK TO SEE IF MERGE OF STATIC DISTRIBUTION TABLE AND OVER-RIDES   03310000
*  RESULTS IN A GOOD DISTRIBUTION TABLE                                03320000
ACT11    DS    0H                                                      03330000
         L     R10,USRVDIST        GET ADDR OF STATIC DIST TABLE       03340000
         USING DISTDSCT,R10        DEFINE BASE FOR STATIC DIST TABLE   03350000
         LA    R6,WSDIST           GET ADDR OF WS DIST TABLE           03360000
         LA    R7,DISTMAX          SET LOOP COUNTER                    03370000
ACT12    DS    0H                  MERGE LOOP                          03380000
         CLC   DISTAPPL,=CL8'NULLAPPL' IS STATIC APPLID NULL?          03390000
         BE    ACT13               YES, SKIP                           03400000
         L     R8,0(,R6)           GET WS DIST VALUE                   03410000
         LTR   R8,R8               IS IT NEGATIVE                      03420000
         BNM   ACT13               NO, SKIP                            03430000
*  SLOT IN STATIC TABLE NOT OVER-RIDDEN, MOVE STATIC ENTRY TO WS       03440000
         MVC   0(L'WSDIST,R6),DISTHIGH  MOVE STATIC TO WS DIST ENTRY   03450000
ACT13    DS    0H                                                      03460000
         CLC   DISTAPPL,=CL8'NULLAPPL' IS STATIC APPLID NULL?          03470000
         BNE   ACT14               NO, SKIP                             03480000
         L     R8,0(,R6)           GET WS DIST VALUE                    03490000
         LTR   R8,R8               IS WS DIST VALUE NEGATIVE?           03500000
         BM    ACT14               YES, SKIP                            03510000
*  NULL ENTRY IN STATIC TABLE OVER-RIDDEN, ERROR                        03520000
         B     PARMERR             ERROR                                03530000
ACT14    DS    0H                                                       03540000
         L     R8,0(R6)            GET WS DIST VALUE                    03550000
         C     R8,=F'100'          IS IT 100?                           03560000
         BNE   ACT15               NO, SKIP                             03570000
         OI    WSSWITCH,WS100FND   YES, SET SWITCH                      03580000
ACT15    DS    0H                                                       03590000
         CLC   DISTAPPL,=CL8'NULLAPPL'  IS STATIC APPLID NULL?          03600000
         BNE   ACT16               NO, SKIP                             03610000
         L     R8,0(,R6)           GET WS DIST VALUE                    03620000
         LTR   R8,R8               IS IT NEGATIVE?                      03630000
         BNM   ACT16               NO, SKIP                             03640000
*  NULL ENTRY IN STATIC TABLE NOT OVER-RIDDEN, SET TO 100 IN WS         03650000
         MVC   0(L'WSDIST,R6),=F'100' SET WS DIST VALUE TO 100          03660000
ACT16    DS    0H                                                       03670000
         LA    R6,L'WSDIST(,R6)    POINT TO NEXT WS DIST ENTRY          03680000
         LA    R10,DISTLEN(,R10)   POINT TO NEXT STATIC DIST ENTRY      03690000
         BCT   R7,ACT12            LOOP DISTMAX TIMES                   03700000
*  IF WS100FND NOT ON, ERROR                                            03710000
         TM    WSSWITCH,WS100FND   IS WS100FND ON?                      03720000
         BZ    PARMERR             NO, ERROR                            03730000
*  WS COPY OF DIST TABLE IS GOOD, MOVE IT TO STATIC COPY IN WS          03740000
         L     R10,USRVDIST        GET ADDR OF STATIC DIST TABLE        03750000
         LA    R6,WSDIST           GET ADDR OF WS DIST TABLE            03760000
         LA    R7,DISTMAX          SET LOOP COUNTER                     03770000
ACT17    DS    0H                  COPY DIST TABLE LOOP                 03780000
         MVC   DISTHIGH,0(R6)      MOVE WS DIST ENTRY TO STATIC ENTRY   03790000
         LA    R6,L'WSDIST(,R6)    POINT TO NEXT WS DIST ENTRY          03800000
         LA    R10,DISTLEN(,R10)   POINT TO NEXT STATIC DIST ENTRY      03810000
         BCT   R7,ACT17            LOOP DISTMAX TIMES                   03820000
         DROP  R3                                                       03830000
         DROP  R10                                                      03840000
         DROP  R11                                                      03850000
ACTRET   DS    0H                                                       03860000
         RETURN (14,12),,RC=0      RETURN                               03870000
         SPACE 1                                                        03880000
PARMERR  DS    0H                                                       03890000
         WTO   'ERROR DETECTED IN USER DATA, USER DATA IGNORED',       X03900000
               ROUTCDE=(10),       SYSTEM ERROR INFORMATION            X03910000
               DESC=(12),          IMPORTANT INFORMATION MESSAGE       X03920000
               LINKAGE=BRANCH      USE BRANCH ENTRY                     03930000
         B     ACTRET              RETURN                               03940000
         TITLE 'SAMPLE USERVAR EXIT TO DISTRIBUTE LOGON REQUESTS TO MULX03950000
               TIPLE IMS SYSTEMS - EXIT DEACTIVATION'                   03960000
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * 03970000
*                                                                     * 03980000
*  IF STORAGE OBTAINED BY EXIT ACTIVATION OR REPLACE ROUTINES THEN    * 03990000
*    FREE STORAGE OBTAINED                                            * 04000000
*  RETURN                                                             * 04010000
*                                                                     * 04020000
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * 04030000
         SPACE 1                                                        04040000
DEACT    DS    0H                  EXIT DEACTIVATION ROUTINE            04050000
         USING PARM1,R2            DEFINE BASE FOR DEACT PARM LIST      04060000
         LR    R2,R1               LOAD BASE FOR DEACT PARM LIST        04070000
         L     R2,USERADDR         GET ADDR OF USER FIELD               04080000
         DROP  R2                  DROP BASE FOR DEACT PARM LIST        04090000
         USING USERFLD,R2          DEFINE BASE FOR USER FIELD AREA      04100000
         L     R3,STORADDR         GET ADDR OF OBTAINED STORAGE         04110000
         DROP  R2                  DROP BASE FOR USER FIELD AREA        04120000
         LTR   R3,R3               IS ADDR ZERO (NO STORAGE OBTAINED)?  04130000
         BZ    DEACTRET            YES, RETURN                         04140000
         L     R2,WRKSPCLN         GET SIZE OF ACQUIRED STORAGE        04150000
         STORAGE RELEASE,COND=YES, FREE OBTAINED STORAGE              X04160000
               LENGTH=(2),ADDR=(3)                                     04170000
         LTR   R15,R15             DID STORAGE RELEASE WORK?           04180000
         BZ    DEACTRET            YES, RETURN                         04190000
         SPACE 1                                                       04200000
         WTO   ’STORAGE RELEASE FAILED IN USERVAR EXIT’,              X04210000
               ROUTCDE=(10),       SYSTEM ERROR INFORMATION           X04220000
               DESC=(12),          IMPORTANT INFORMATION MESSAGE      X04230000
               LINKAGE=BRANCH      USE BRANCH ENTRY                    04240000
         SPACE 1                                                       04250000
DEACTRET DS    0H                                                      04260000
         RETURN (14,12),,RC=0      RETURN                              04270000
         TITLE ’SAMPLE USERVAR EXIT TO DISTRIBUTE LOGON REQUESTS TO MULX04280000
               TIPLE IMS SYSTEMS - MACRO DEFINITION’                   04290000
         MACRO                                                         04300000
         VARDEF &NAME,&P1,&P2,&P3,&P4,&P5,&P6,&P7,&P8,&P9,&P10,       X04310000
               &LAST=NO                                                04320000
         GBLA  &NMACRO                                                 04330000
         LCLA  &LCTR,&DPCT(10),&HIGHVAL,&TCTR                          04340000
         LCLB  &GOODEND                                                04350000
         LCLC  &DNAM(10),&VARNAM,&TLABEL                               04360000
&NMACRO  SETA  &NMACRO+1           INCREMENT COUNT OF VARDEF MACROS    04370000
         AIF   (&NMACRO LE 5).GOOD1                                    04380000
         MNOTE 16,’MORE THAN 5 VARDEF MACROS SPECIFIED, MACRO IGNORED’  04390000
         MEXIT                                                         04400000
.GOOD1   ANOP                                                          04410000
&VARNAM  SETC  ’ ’                 INITIALIZE USERVAR NAME             04420000
&LCTR    SETA     1                                                    04430000
.LOOP1   ANOP                                                          04440000
&DNAM(&LCTR) SETC ’NULLAPPL’       INITIALIZE APPLID                   04450000
&DPCT(&LCTR) SETA 100              INITIALIZE DIST PCT TO 100          04460000
&LCTR    SETA  &LCTR+1             INCREMENT LOOP COUNTER              04470000
         AIF   (&LCTR LE 10).LOOP1 LOOP                                04480000
         AIF   (T’&NAME NE ’O’).GOOD2                                  04490000
         MNOTE 16,’USERVAR NAME PARAMETER MUST BE SPECIFIED’           04500000
         MEXIT                                                         04510000
.GOOD2   ANOP                                                          04520000
&VARNAM  SETC  ’&NAME’             SET USERVAR NAME                    04530000
         AIF   (T’&P1 NE ’O’).GOOD3                                    04540000
         MNOTE 16,’2ND PARAMETER NOT SPECIFIED’                        04550000
         MEXIT                                                         04560000
.GOOD3   ANOP                                                          04570000
         AIF   (N’&P1 EQ 2).GOOD4                                      04580000
         MNOTE 16,’2ND PARAMETER MUST HAVE TWO VALUES SPECIFIED’       04590000
         MEXIT                                                         04600000
.GOOD4   ANOP                                                          04610000
&DNAM(1) SETC ’&P1(1)’             SET APPLID                          04620000
&DPCT(1) SETA &P1(2)               SET PERCENTAGE                      04630000
         AIF   (T’&P2 EQ ’O’).GENERAT                                  04640000
&DNAM(2) SETC ’&P2(1)’             SET APPLID                          04650000
&DPCT(2) SETA &P2(2)               SET PERCENTAGE                      04660000
         AIF   (T’&P3 EQ ’O’).GENERAT                                  04670000
&DNAM(3) SETC ’&P3(1)’             SET APPLID                          04680000
&DPCT(3) SETA &P3(2)               SET PERCENTAGE                      04690000
         AIF   (T’&P4 EQ ’O’).GENERAT                                  04700000
&DNAM(4) SETC ’&P4(1)’             SET APPLID                          04710000
&DPCT(4) SETA &P4(2)               SET PERCENTAGE                      04720000
         AIF   (T’&P5 EQ ’O’).GENERAT                                  04730000
&DNAM(5) SETC ’&P5(1)’             SET APPLID                          04740000
&DPCT(5) SETA &P5(2)               SET PERCENTAGE                      04750000
         AIF   (T’&P6 EQ ’O’).GENERAT                                  04760000
&DNAM(6) SETC ’&P6(1)’             SET APPLID                          04770000
&DPCT(6) SETA &P6(2)               SET PERCENTAGE                      04780000
```

```
         AIF   (T’&P7 EQ ’O’).GENERAT                                   04790000
&DNAM(7) SETC ’&P7(1)’             SET APPLID                           04800000
&DPCT(7) SETA &P7(2)               SET PERCENTAGE                       04810000
         AIF   (T’&P8 EQ ’O’).GENERAT                                   04820000
&DNAM(8) SETC ’&P8(1)’             SET APPLID                           04830000
&DPCT(8) SETA &P8(2)               SET PERCENTAGE                       04840000
         AIF   (T’&P9 EQ ’O’).GENERAT                                   04850000
&DNAM(9) SETC ’&P9(1)’             SET APPLID                           04860000
&DPCT(9) SETA &P9(2)               SET PERCENTAGE                       04870000
         AIF   (T’&P10 EQ ’O’).GENERAT                                  04880000
&DNAM(10) SETC ’&P10(1)’           SET APPLID                           04890000
&DPCT(10) SETA &P10(2)             SET PERCENTAGE                       04900000
.GENERAT ANOP                                                           04910000
&LCTR SETA     1                   INITIALIZE LOOP COUNTER              04920000
&HIGHVAL SETA  0                                                        04930000
.LOOP2   ANOP                                                           04940000
         AIF   (&DPCT(&LCTR) GE &HIGHVAL).GOOD5                         04950000
         MNOTE 16,’DISTRIBUTION PERCENTAGES ARE NOT INCREASING’         04960000
         MEXIT                                                          04970000
.GOOD5   ANOP                                                           04980000
&HIGHVAL SETA  &DPCT(&LCTR)        SET NEW HIGH VALUE                   04990000
         AIF   ((&DPCT(&LCTR) GE 0) AND                                X05000000
               (&DPCT(&LCTR) LE 100)).GOOD6                             05010000
         MNOTE 16,’DISTRIBUTION PERCENTAGES NOT BETWEEN 0 AND 100’      05020000
         MEXIT                                                          05030000
.GOOD6   ANOP                                                           05040000
         AIF   ((&DPCT(&LCTR) NE 100) OR                               X05050000
               (’&DNAM(&LCTR)’ EQ ’NULLAPPL’)).GOOD7                    05060000
&GOODEND SETB  1                                                        05070000
.GOOD7   ANOP                                                           05080000
&LCTR    SETA  &LCTR+1             INCREMENT LOOP COUNTER               05090000
         AIF   (&LCTR LE 10).LOOP2 LOOP                                 05100000
         AIF   (&GOODEND).GOOD8                                         05110000
         MNOTE 16,’100 MUST BE SPECIFIED FOR AT LEAST ONE ENTRY’        05120000
         MEXIT                                                          05130000
.GOOD8   ANOP                                                           05140000
         PUSH  PRINT                                                    05150000
         PRINT GEN                                                      05160000
&TLABEL  SETC  ’USRVTBL’.’&NMACRO’ BUILD LABEL NAME                     05170000
&TLABEL  DS    0F                                                       05180000
         DC    CL8’&VARNAM’        USERVAR NAME TO BE CONTROLLED        05190000
&TLABEL  SETC  ’DISTTBL’.’&NMACRO’ BUILD LABEL NAME                     05200000
         DC    A(&TLABEL)          ADDR OF DIST TABLE                   05210000
&TCTR    SETA  &NMACRO+1                                                05220000
&TLABEL  SETC  ’USRVTBL’.’&TCTR’   BUILD LABEL NAME                     05230000
         AIF   (’&LAST’ EQ ’YES’).GOOD9                                 05240000
         DC    A(&TLABEL)          ADDR OF NEXT USERVAR ENTRY           05250000
         AGO   .GOODA                                                   05260000
.GOOD9   ANOP                                                           05270000
         DC    A(0)                ADDR OF NEXT USERVAR ENTRY           05280000
.GOODA   ANOP                                                           05290000
&TLABEL  SETC  ’DISTTBL’.’&NMACRO’ BUILD LABEL NAME                     05300000
&LCTR    SETA  1                                                        05310000
.LOOP3   ANOP                                                           05320000
         AIF   (&LCTR EQ 1).GOODB                                       05330000
&TLABEL  SETC  ’ ’                                                      05340000
.GOODB   ANOP                                                           05350000
&TLABEL  DC    F’&DPCT(&LCTR)’     CUMMULATIVE DIST PERCENTAGE          05360000
         DC    CL8’&DNAM(&LCTR)’   APPLID                               05370000
&LCTR    SETA  &LCTR+1             INCREMENT LOOP CTR                   05380000
         AIF   (&LCTR LE 10).LOOP3                                      05390000
         POP   PRINT                                                    05400000
         MEND                                                           05410000
         TITLE ’SAMPLE USERVAR EXIT TO DISTRIBUTE LOGON REQUESTS TO MULX05420000
               TIPLE IMS SYSTEMS - TABLE DEFINITIONS’                   05430000
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * 05440000
*                                                                     *
*  THE VARDEF MACRO IS USED TO SPECIFY THE USERVARS THAT THIS EXIT IS *
*  TO CONTROL AND HOW THE USERVAR TRANSLATIONS ARE TO BE HANDLED      *
*  (REAL APPLIDS AND DISTRIBUTION).  UP TO 5 DIFFERENT VARDEF MACROS  *
*  CAN BE SPECIFIED.  THE VARDEF MACROS SHOULD BE CONTIGUOUS (NOT     *
*  ABSOLUTELY REQUIRED).  THE VARDEF PARAMETER CONTAINS UP TO 11      *
*  POSITIONAL PARAMETERS AND ONE KEYWORD PARAMETER (LAST=YES - MUST   *
*  BE SPECIFIED ON THE LAST VARDEF MACRO).  THE FORMAT OF VARDEF      *
*  MACRO IS AS FOLLOWS:                                               *
*                                                                     *
*      VARDEF NAME,D1,D2,D3,D4,D5,D6,D7,D8,D9,D10,LAST=YES            *
*                                                                     *
*  WHERE                                                              *
*                                                                     *
*    NAME:  USERVAR NAME TO BE TRANSLATED (REQUIRED)                  *
*                                                                     *
*    D1 THROUGH D10:  APPLID AND CUMULATIVE PERCENTAGE, ENCLOSED IN   *
*                     PARENTHESES.  D1 IS REQUIRED.  D2 THROUGH D10   *
*                     ARE OPTIONAL.  THE PERCENTAGE CAN RANGE FROM 0  *
*                     TO 100.  CUMULATIVE PERCENTAGES MUST INCREASE   *
*                     AND THE LAST PARAMETER SPECIFIED MUST HAVE A    *
*                     CUMULATIVE PERCENTAGE OF 100.                   *
*                                                                     *
*                                                                     *
*    LAST=YES:  MUST (AND ONLY SHOULD) BE SPECIFIED FOR THE LAST      *
*               VARDEF MACRO                                          *
*                                                                     *
*  EXAMPLES:                                                          *
*                                                                     *
*  1.  ASSUME THAT LOGON REQUESTS AGAINST USERVAR IMSPLEX1 ARE TO BE  *
*      DISTRIBUTED 50 PERCENT TO IMS1 AND 50 PERCENT TO IMS2.  ONLY   *
*      ONE USERVAR IS TO BE MANAGED BY THIS EXIT.  THE APPROPRIATE    *
*      VARDEF WOULD BE:                                               *
*                                                                     *
*       VARDEF IMSPLEX1,(50,IMS1),(100,IMS2)                          *
*                                                                     *
*  2.  ASSUME THAT LOGON REQUESTS AGAINST USERVAR IMSPLEX1 ARE TO BE  *
*      DISTRIBUTED 40 PERCENT TO IMS1, 30 PERCENT TO IMS2, 20 PERCENT *
*      TO IMS3, AND 10 PERCENT TO IMS4.  ONLY ONE USERVAR IS TO BE    *
*      MANAGED BY THIS EXIT.  THE VARDEF MACRO REQUIRED TO IMPLEMENT  *
*      THIS DISTRIBUTION COULD BE CODED MANY WAYS.  AMONG THE MANY    *
*      WAYS ARE:                                                      *
*                                                                     *
*       VARDEF IMSPLEX1,(40,IMS1),(70,IMS2),(90,IMS3),(100,IMS4)      *
*       VARDEF IMSPLEX1,(10,IMS4),(30,IMS3),(60,IMS2),(100,IMS1)      *
*       VARDEF IMSPLEX1,(20,IMS3),(60,IMS1),(70,IMS4),(100,IMS2)      *
*                                                                     *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
         SPACE 1
         VARDEF IMSPLEX,(IMS1,50),(IMS2,100)
         SPACE 1
         VARDEF IMSPLEX2,(IMS100,40),(IMS200,70),(IMS300,90),          X
               (IMS400,100),LAST=YES
         SPACE 1
WRKSPCLN DC    A((USRVMAX*USRVLEN)+(DISTMAX*DISTLEN*USRVMAX)+WORKLEN)
WORKLEN  EQU   WSLEN+DISTMAX*L'WSDIST
HIOFF    DC    X'7FFFFFFF'
HEX04    DC    H'04'               HEX 04
HEX10    DC    H'16'               HEX 10
HEX18    DC    H'24'               HEX 18
HEX20    DC    H'32'               HEX 20
HEX40    DC    H'64'               HEX 40
         SPACE 1
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*                                                                     *
*  THE VALUE OF VERSION SHOULD BE CHANGED WHEN THE VERSION OF THIS    *
*  EXIT IS CHANGED.  IT SHOULD AGREE WITH THE EYECATCHER IN THE       * 06110000
*  SAVE MACRO.  THE VERSION WILL BE DISPLAYED IN THE VTAM DISPLAY     * 06120000
*  EXIT COMMAND.  USING UNIQUE VERSION AND EYECATCHER VALUES WILL     * 06130000
*  AVOID PROBLEM DETERMINATION CONFUSION.                             * 06140000
*                                                                     * 06150000
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * 06160000
         SPACE 1                                                        06170000
VERSION  DC    CL4’V001’           VERSION ID                           06180000
         SPACE 1                                                        06190000
TRTABLE  DC    X’FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF’  X’00’ - X’0F’       06200000
         DC    X’FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF’  X’10’ - X’1F’       06210000
         DC    X’FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF’  X’20’ - X’2F’       06220000
         DC    X’FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF’  X’30’ - X’3F’       06230000
         DC    X’FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF’  X’40’ - X’4F’       06240000
         DC    X’FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF’  X’50’ - X’5F’       06250000
         DC    X’FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF’  X’60’ - X’6F’       06260000
         DC    X’FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF’  X’70’ - X’7F’       06270000
         DC    X’FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF’  X’80’ - X’8F’       06280000
         DC    X’FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF’  X’90’ - X’9F’       06290000
         DC    X’FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF’  X’A0’ - X’AF’       06300000
         DC    X’FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF’  X’B0’ - X’BF’       06310000
         DC    X’FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF’  X’C0’ - X’CF’       06320000
         DC    X’FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF’  X’D0’ - X’DF’       06330000
         DC    X’FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF’  X’E0’ - X’EF’       06340000
         DC    X’00000000000000000000FFFFFFFFFFFF’  X’F0’ - X’FF’       06350000
         SPACE 1                                                        06360000
         LTORG                                                          06370000
         TITLE ’SAMPLE USERVAR EXIT TO DISTRIBUTE LOGON REQUESTS TO MULX06380000
               TIPLE IMS SYSTEMS - DSECT DEFINITIONS’                   06390000
USRVDSCT DSECT ,                   USERVAR NAME ENTRY (5 OCCURRENCES)   06400000
USRVNAME DS    CL8                 USERVAR NAME                         06410000
USRVDIST DS    A                   ADDR OF DISTRIBUTION TABLE           06420000
USRVNEXT DS    A                   ADDR OF NEXT ENTRY                   06430000
USRVLEN  EQU   *-USRVDSCT          LENGTH OF USERVAR NAME ENTRY         06440000
USRVMAX  EQU   5                   MAXIMUM NUMBER OF USERVAR ENTRIES    06450000
         SPACE 1                                                        06460000
DISTDSCT DSECT ,                   USERVAR DIST TABLE (10 OCCURRENCES)  06470000
DISTHIGH DS    F                   HIGH RANGE FOR APPLID                06480000
DISTAPPL DS    CL8                 APPLID TO ASSIGN                     06490000
DISTLEN  EQU   *-DISTDSCT          LENGTH OF DIST TABLE ENTRY           06500000
DISTMAX  EQU   10                  MAXIMUM NUMBER OF DIST ENTRIES       06510000
         SPACE 1                                                        06520000
WORKSTOR DSECT ,                   WORKING STORAGE (USER DATA PROC)     06530000
WSDOUBLE DS    D                   DOUBLE WORD WORK AREA                06540000
WSSWITCH DS    XL1                 SWITCHES                             06550000
WS100FND EQU   X’01’               VALUE OF 100 FOUND                   06560000
         DS    XL1                 FILLER                               06570000
WSOLDHI  DS    H                   HALF WORD WORK AREA                  06580000
WSUSRVAR DS    CL8                 USER DATA USERVAR NAME               06590000
WSLEN    EQU   *-WORKSTOR          LENGTH OF WS BASE SECTION            06600000
WSDIST   DS    0F                  USER DATA DISTRIBUTION ENTRY         06610000
*                                  (DISTMAX ENTRIES)                    06620000
         SPACE 1                                                        06630000
PARM1    DSECT ,                   PARM LIST DEFINITION FOR ACTIVATE,   06640000
*                                  DEACTIVATE, AND REPLACE              06650000
USERADDR DS    A                   ADDR OF USER FIELD                   06660000
         DS    XL8                 RESERVED                             06670000
PARMADDR DS    A                   ADDR OF USER PARMS                   06680000
         SPACE 1                                                        06690000
USERFLD  DSECT ,                   USER FIELD DSECT                     06700000
STORADDR DS    A                   ADDR OF OBTAINED STORAGE             06710000
VERID    DS    CL4                 EXIT VERSION ID                      06720000
         SPACE 1                                                        06730000
PARM2    DSECT ,                   PARM LIST DEFINITION FOR INVOKE      06740000
USERADD1 DS    A                   ADDR OF USER FIELD                   06750000
```

```
         DS    XL8                 RESERVED                             06760000
INVOKEA  DS    A                   ADDR OF INVOKE FLAGS                 06770000
USRVPRMS DS    A                   ADDR OF USERVAR PARMS                06780000
         SPACE 1                                                        06790000
USRVPARM DSECT ,                   USERVAR PARMS                        06800000
OLUNAME  DS    CL8                 OLU NAME                             06810000
OLUADR   DS    XL4                 OLU SUBAREA ADDR                     06820000
SESSADDR DS    A                   ADDR OF SESSION PARTNERS TABLE       06830000
COSNAME  DS    CL8                 COS NAME FOR SESSION                 06840000
GENERIC  DS    CL8                 USERVAR NAME                         06850000
FLAGS    DS    XL1                 FLAGS                                06860000
TRANSLAT EQU   X'20'               USERVAR TRANSLATED                   06870000
VOLATILE EQU   X'18'               VOLATILE USERVAR                     06880000
         DS    XL3                 RESERVED                             06890000
TRANVAL  DS    CL8                 TRANSLATED USERVAR VALUE             06900000
         DS    CL8                 RESERVED                             06910000
         END                                                            06920000
```

# Appendix D. Parallel Sysplex Publications

Individuals planning a migration to Parallel Sysplex will want to have the following publications available.

- *OS/390 Setting Up a Sysplex*, GC28-1779

  This book documents the Administrative Data Utility, explains Automatic Restart Management, and contains other information on implementing a Parallel Sysplex.

- *IMS/ESA Data Sharing in a Parallel Sysplex*, SG24-4303

  This redbook documents IMS data sharing. It includes implementation, operations, recovery, and performance information.

- *IMS/ESA Sysplex Data Sharing: An Implementation Case Study*, SG24-4831

  This redbook documents the implementation of IMS/ESA Version 5 data sharing at an installation.

- *IMS/ESA Multiple Systems Coupling in a Parallel Sysplex*, SG24-4750

  This redbook documents the use of IMS MSC in a Parallel Sysplex.

- *MVS/ESA Programming: Sysplex Services Guide*, GC28-1495

  This book describes the services that MVS provides to enable multisystem applications and subsystems, such as IMS and IRLM, to use Parallel Sysplex facilities. The information is much more detailed than normally required for use of IMS. For those who like to know what′s going on in the system, this will provide some insight.

- *OS/390 MVS Parallel Sypslex Configuration Volume 1: Overview*, SG24-2075

  This redbook contains an overview of Parallel Sysplex and its use by various subsystems, such as IMS.

- *OS/390 MVS Parallel Sypslex Configuration Volume 2: Cookbook*, SG24-2076

  This redbook contains information on the use of Parallel Sysplex by various subsystems. It includes information on CF use and structures.

- *OS/390 MVS Parallel Sypslex Configuration Volume 3: Connectivity*, SG24-2077

  This redbook contains information on the connections, such as channels, ESCON, networks, consoles, and Sysplex Timers in a Parallel Sysplex.

# Appendix E. Migration Plan Task List

The following are sample lists of tasks that might be included in a plan for migrating an IMS system to data sharing in a Parallel Sysplex. They include tasks to migrate an IMS TM system to generic resources and shared queues. The lists are intended to include tasks that could be in your plan. You can identify additional tasks, and there might be some tasks that are not required in your plan.

The task lists are broken into three parts corresponding to the three phases of the migration process identified in Chapter 2, "Plan Development" on page 5. The planning and preparation phases prepare an installation to implement the capabilities. The implementation phase introduces the capabilities into the system. There are four lists for this phase. The first three are for implementing data sharing, generic resources, and shared queues environments. These lists do not include the implementation of a second IMS system using these facilities. They are used to implement an environment where a single IMS system is using data sharing, generic resources, or shared queues. The last list in the implementation phase is for adding the capabilities to a second system.

In the lists, the column labeled ENVIR indicates whether the task would be required for IMS Transaction Manager only (TM) or both DBCTL and TM (BOTH).

## E.1 Planning Phase

*Table 10. Task List*

| TASK | REFERENCE | ENVIR |
|---|---|---|
| Evaluate existing environment<br>• Document reasons for migrating<br>• Document current workload<br>• Identify heavy resource users<br>• Document service level commitments<br>• Document connections to other systems<br>• Document security requirements<br>• Document online and batch schedules<br>• Document online and batch dependencies<br>• Identify user exits and determine need to update<br>• Identify critical applications<br>• Identify non-sharable resources<br>• Identify potential performance problems | 2.1.1.1 | BOTH |
| Define target configuration<br>• Identify MVS/IMS/DB2/CICS/CF configuration<br>• Identify IMS affinities<br>• Identify applications and databases to be partitioned (if any)<br>• Identify network connections<br>• Identify remote system connections<br>• Identify security requirements | 2.1.1.2, Chapter 6, 7.2, Chapter 8, 8.2, Chapter 10, and 16.1 | BOTH |
| Define degraded mode environment<br>• Perform CFIA for critical components<br>• Prioritize applications for degraded mode processing<br>• Identify applications to run in degraded mode<br>• Identify network terminals and connections to be reconnected | 2.1.1.3, and Chapter 4 | BOTH |
| Develop the plan<br>• Select project management tool (if any)<br>• Identify tasks<br>• Assign responsibilities<br>• Determine schedules | 2.1.1.4 | BOTH |

## E.2 Preparation Phase

*Table 11 (Page 1 of 2). Task List*

| TASK | REFERENCE | ENVIR |
|---|---|---|
| Establish naming conventions | 5.1 | BOTH |
| Develop plan to handle IMS system data sets<br>• Categorize IMS system data sets as unique, shared, or cloned<br>• Assign data set names to IMS system data sets<br>• Develop plan to copy/rename/create system data sets in target environment | 5.2.1 and Appendix B | BOTH |
| Develop plan for handling applications and databases in target environment<br>• Cloned applications<br>• Cloned databases<br>• Partitioned applications<br>• Partitioned databases<br>• Cloned application data sets<br>• Databases that cannot be shared | Chapter 6 | BOTH |
| Develop plan for partitioning IMS network in target environment | Chapter 8 and 8.2 | TM |
| Develop plan for balancing IMS TM workload | Chapter 9 | TM |
| Develop plan for connecting remote systems to target systems | 16.1 | TM |
| Develop plan for implementing security in target environment | 16.2 | BOTH |
| Choose CRCs for clones | 18.9 | BOTH |
| Review and update user exit routines | 16.5 | BOTH |
| Review and update JCL<br>• IMS JOBs<br>• IMS PROCs<br>• DBRC skeletal JCL<br>• Other DB support JCL<br>• Application JCL | Chapter 17 | BOTH |
| Review and update operational procedures<br>• System startup/shutdown procedures<br>• System recovery procedures<br>• Database recovery procedures<br>• IMS log management procedures<br>• Job scheduling procedures<br>• BMP restart procedures | Chapter 18 | BOTH |

*Table 11 (Page 2 of 2). Task List*

| TASK | REFERENCE | ENVIR |
|---|---|---|
| Review and update support procedures<br>• IMS maintenance<br>• IMS system definition<br>• Security definition<br>• MFS format generation<br>• DBD, PSB, ACB generation<br>• Online change<br>• Application maintenance<br>• Change control<br>• Database reorganization<br>• Database backup and recovery | Chapter 18 | BOTH |
| Review and update IMS MTO/system operator procedures<br>• System startup and shutdown<br>• System recovery<br>• Resource management<br>• System performance monitoring<br>• Online change | Chapter 18 | BOTH |
| Review and update automated operations procedures | 18.2 | BOTH |
| Review and update disaster recovery plans | 20.14 | BOTH |
| Develop plan to enable data sharing | Chapter 11 | |
| Design coupling facility environment<br>• Determine CF structure sizes<br>• Choose placement of CF structures<br>• Establish procedures for change of CF structure sizes and placement<br>• Establish procedures for CF structure recovery | Chapter 15 and 15.1 | BOTH |
| Evaluate potential performance problems in target environment<br>• Data sharing considerations<br>• Processor speed<br>• Workload balancing | 11.9 | BOTH |
| Develop plan to implement degraded mode processing<br>• Establish procedures for implementing degraded mode processing | 4.8 | BOTH |

## E.3 Implementation Phase

### E.3.1 Data Sharing Environment Implementation

*Table 12. Task List*

| TASK | REFERENCE | ENVIR |
|---|---|---|
| Change RECONs to SHARECTL | 11.3 | TM |
| Register databases at SHARELVL(1) | 11.3 | BOTH |
| Update IMS system definition for data sharing | 11.1 | BOTH |
| Specify DISP=SHR for database data sets | 11.4 | BOTH |
| Specify SHAREOPTION(3 3) for VSAM database data sets | 11.4 | BOTH |
| Add IRLM subsystem names to IEFSSNxx | 11.2.1 | BOTH |
| Implement IRLM as lock manager with SCOPE=LOCAL | 11.2 | BOTH |
| Register databases at SHARELVL(2) | 11.3 | BOTH |
| Define coupling facility structures | Chapter 15 and 15.1 | BOTH |
| Include a CFNAMES statement in DFSVSMxx member and in DFSVSAMP for batch BLDS jobs | 11.5 | BOTH |
| Implement IRLM as lock manager with SCOPE=NODISCON | 11.2.3 | BOTH |
| Register databases at SHARELVL(3) | 11.3 | BOTH |

### E.3.2 Generic Resources Environment Implementation

*Table 13. Task List*

| TASK | REFERENCE | ENVIR |
|---|---|---|
| Define coupling facility structure | Chapter 13 | TM |
| Specify GRSNAME IMS execution parameter | Chapter 13 | TM |
| Specify GRNAME on the LUADD statement for APPC/IMS | Chapter 13 | TM |

## E.3.3 Shared Queues Environment Implementation

*Table 14. Task List*

| TASK | REFERENCE | ENVIR |
|---|---|---|
| Update program properties table | 12.1 | TM |
| Add structure definitions to CFRM policy | 12.2, Chapter 15, and 15.1 | TM |
| Define log streams in logger policy | 12.3 | TM |
| Define structure checkpoint data sets | 12.4 | TM |
| Define structure recovery data sets | 12.5 | TM |
| Create CQS procedure | 12.6 | TM |
| Create BPE configuration PROCLIB member | 12.7 | TM |
| Create CQSIPxxx PROCLIB member | 12.8 | TM |
| Create CQSSLxxx PROCLIB member | 12.9 | TM |
| Create CQSSGxxx PROCLIB member | 12.10 | TM |
| Define security for CQS structures | 12.11 and 16.2 | TM |
| Update ARM policy | 12.12 and Chapter 14 | TM |
| Create DFSSQxxx PROCLIB member | 12.13 | TM |
| Update DFSVSMxx PROCLIB member for traces | 12.14 | TM |
| Update IMS procedure in invoke shared queues | 12.15 | TM |

## E.3.4 Implementation of Second System (Clone)

*Table 15. Task List*

| TASK | REFERENCE | ENVIR |
|---|---|---|
| Develop clone of IMS system |  | BOTH |
| - Develop unique IMS system data sets for clone | 5.2.1 and Appendix B | BOTH |
| - Develop system definition for clone | Chapter 10 and 11.1 | BOTH |
| - Establish execution parameters for clone |  | BOTH |
| - Develop PROCs for clone | 12.16 | BOTH |
| - Develop archive skeletal JCL for clone, if necessary | 17.3 | BOTH |

# Appendix F. Special Notices

This publication is intended to help users to identify steps, document, and what to do if there are failures when planning to migrate an IMS online system into Parallel Sysplex environment. It is also intended to help IBM representatives to have a better understand of IMS in parallel sysplex environment so that he or she will be able better to serve customers. The information in this publication is not intended as the specification of any programming interfaces that are provided by IMS/ESA product. See the PUBLICATIONS section of the IBM Programming Announcement for IMS/ESA Version 6.1 for more information about what publications are considered to be product documentation.

References in this publication to IBM products, programs or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM′s product, program, or service may be used. Any functionally equivalent program that does not infringe any of IBM′s intellectual property rights may be used instead of the IBM product, program or service.

Information in this book was developed in conjunction with use of the equipment specified, and is limited in application to those specific hardware and software products and levels.

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

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM Corporation, Dept. 600A, Mail Drop 1329, Somers, NY 10589 USA.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The information contained in this document has not been submitted to any formal IBM test and is distributed AS IS. The information about non-IBM (″vendor″) products in this manual has been supplied by the vendor and IBM assumes no responsibility for its accuracy or completeness. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer′s ability to evaluate and integrate them into the customer′s operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

Any pointers in this publication to external Web sites are provided for convenience only and do not in any manner serve as an endorsement of these Web sites.

Any performance data contained in this document was determined in a controlled environment, and therefore, the results that may be obtained in other operating environments may vary significantly. Users of this document should verify the applicable data for their specific environment.

Reference to PTF numbers that have not been released through the normal distribution process does not imply general availability. The purpose of including these reference numbers is to alert IBM customers to specific information relative to the implementation of the PTF when it becomes available to each customer according to the normal IBM PTF distribution process.

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

ACF/VTAM
APPN
AS/400
AT
CICS
CICS/ESA
CICS/MVS
CICSPlex
DB2
DFSMS
ESCON
Extended Services
Hiperspace
IBM
IMS
IMS/ESA
MQ
MQSeries
MVS/DFP
MVS/ESA
NetView
OS/390
Parallel Sysplex
PR/SM
RACF
RMF
RS/6000
S/390
SP
System/390
VTAM

The following terms are trademarks of other companies:

C-bus is a trademark of Corollary, Inc. in the United States and/or other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Micorsystems, Inc. in the United States and/or other countries.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States and/or other countries.

PC Direct is a trademark of Ziff Communications Company in the United States and is used by IBM Corporation under license.

ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States and/or other countries.

UNIX is a registered trademark in the United States and/or other countries licensed exclusively through X/Open Company Limited.

SET and the SET logo are trademarks owned by SET Secure Electronic Transaction LLC.

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

# Appendix G. Related Publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

## G.1 International Technical Support Organization Publications

For information on ordering these ITSO publications see "How to Get ITSO Redbooks" on page 243.

- *IMS/ESA Data Sharing in a Parallel Sysplex*, SG24-4303
- *IMS/ESA Sysplex Data Sharing: An Implementation Case Study*, SG24-4831
- *IMS/ESA Multiple Systems Coupling in a Parallel Sysplex*, SG24-4750
- *OS/390 MVS Parallel Sysplex Configuration Volume 1: Overview*, SG24-2075
- *OS/390 MVS Parallel Sysplex Configuration Volume 2: Cookbook*, SG24-2076
- *OS/390 MVS Parallel Sysplex Configuration Volume 3: Connectivity*, SG24-2077

## G.2 Redbooks on CD-ROMs

Redbooks are also available on the following CD-ROMs. Click the CD-ROMs button at `http://www.redbooks.ibm.com/` for information about all the CD-ROMs offered, updates and formats.

| **CD-ROM Title** | **Collection Kit Number** |
|---|---|
| System/390 Redbooks Collection | SK2T-2177 |
| Networking and Systems Management Redbooks Collection | SK2T-6022 |
| Transaction Processing and Data Management Redbooks Collection | SK2T-8038 |
| Lotus Redbooks Collection | SK2T-8039 |
| Tivoli Redbooks Collection | SK2T-8044 |
| AS/400 Redbooks Collection | SK2T-2849 |
| Netfinity Hardware and Software Redbooks Collection | SK2T-8046 |
| RS/6000 Redbooks Collection (BkMgr Format) | SK2T-8040 |
| RS/6000 Redbooks Collection (PDF Format) | SK2T-8043 |
| Application Development Redbooks Collection | SK2T-8037 |

## G.3 Other Publications

These publications are also relevant as further information sources:

- *IMS/ESA Recovery Saver User′s Guide and Reference*, SC26-9191
- *IMS/ESA Common Queue Server Guide and Reference*, SC26-9517
- *MVS/ESA Programming: Sysplex Services Guide*, GC28-1495
- *OS/390 Security Server (RACF) Security Administration Guide*, SC23-3726
- *OS/390 Setting Up a Sysplex*, GC28-1779
- *OS/390 Security Server (RACF) Security Administration Guide*, SC28-1915
- *OS/390 Parallel Sysplex Application Migration*, GC28-1863
- *S/390 PR/SM Planning Guide*, GA22-7236

- *DB2 for OS/390 Version 5 DB2 Data Sharing:Planning and Administration*, SC26-8961

# How to Get ITSO Redbooks

This section explains how both customers and IBM employees can find out about ITSO redbooks, redpieces, and CD-ROMs. A form for ordering books and CD-ROMs by fax or e-mail is also provided.

- **Redbooks Web Site** http://www.redbooks.ibm.com/

  Search for, view, download, or order hardcopy/CD-ROMs redbooks from the redbooks Web site. Also read redpieces and download additional materials (code samples or diskette/CD-ROM images) from this redbooks site.

  Redpieces are redbooks in progress; not all redbooks become redpieces and sometimes just a few chapters will be published this way. The intent is to get the information out much quicker than the formal publishing process allows.

- **E-mail Orders**

  Send orders by e-mail including information from the redbook fax order form to:

| | |
|---|---|
| In United States: | e-mail address: usib6fpl@ibmmail.com |
| Outside North America: | Contact information is in the ″How to Order″ section at this site:<br>http://www.elink.ibmlink.ibm.com/pbl/pbl/ |

- **Telephone Orders**

| | |
|---|---|
| United States (toll free) | 1-800-879-2755 |
| Canada (toll free) | 1-800-IBM-4YOU |
| Outside North America | Country coordinator phone number is in the ″How to Order″ section at this site:<br>http://www.elink.ibmlink.ibm.com/pbl/pbl/ |

- **Fax Orders**

| | |
|---|---|
| United States (toll free) | 1-800-445-9269 |
| Canada | 1-403-267-4455 |
| Outside North America | Fax phone number is in the ″How to Order″ section at this site:<br>http://www.elink.ibmlink.ibm.com/pbl/pbl/ |

This information was current at the time of publication, but is continually subject to change. The latest information may be found at the redbooks Web site.

**IBM Intranet for Employees**

IBM employees may register for information on workshops, residencies, and redbooks by accessing the IBM Intranet Web site at http://w3.itso.ibm.com/ and clicking the ITSO Mailing List button. Look in the Materials repository for workshops, presentations, papers, and Web pages developed and written by the ITSO technical professionals; click the Additional Materials button. Employees may access MyNews at http://w3.ibm.com/ for redbook, residency, and workshop announcements.

## IBM Redbook Fax Order Form

**Please send me the following:**

| Title | Order Number | Quantity |
|---|---|---|
| | | |
| | | |
| | | |
| | | |

First name Last name

Company

Address

City Postal code Country

Telephone number Telefax number VAT number

- Invoice to customer number ____________________

- Credit card number ____________________

Credit card expiration date Card issued to Signature

**We accept American Express, Diners, Eurocard, Master Card, and Visa. Payment by credit card not available in all countries. Signature mandatory for credit card payment.**

# Glossary

**access method**. A technique for moving data between main storage and input/output devices, for example, VSAM, VTAM.

**access method control block (ACB)**. A control block that links an application program (for example, a CICS system) to an access method (for example VSAM or VTAM). In communication with DL/I, an ACB is used only when the underlying access method is VSAM.

**access method services (AMS)**. A utility program for the definition and management of VSAM data sets.

**active IRLM**. The IRLM supporting the active IMS subsystem in an XRF complex.

**active libraries**. The libraries from which IMS draws its execution information when online change is used.

**address space**. A range of up to two gigabytes of contiguous virtual storage addresses that the system creates for the user. Unlike a data space, an address space contains user data and programs, as well as system data and programs, some of which are common to all address spaces. Instructions execute in an address space not a data space.

**APPC/IMS**. A part of IMS TM that uses the CPI communication interface to talk to application programs.

**application control blocks (ACB)**. The control blocks created from the output of DBDGEN and PSBGEN and placed in the ACB library for use during online and DBB region type execution of IMS.

**application identifier (VTAM applid)**. The name by which a logical unit is known in a VTAM network. The CICS applid is specified in the APPLID system initialization parameter.

**application load balancing**. An optional facility that enables an application program to be scheduled into more than one message or batch message region at the same time.

**Application Group Name**. A name that represents a defined of IMS resources (PSBs, transaction names, and logical terminal names).

**APPLID name**. The name of which VTAM knows as an IMS system for establishing sessions. The name is specified in a VTAM APPL definition statement and in the APPLID keyword of the IMS COMM system definition macro.

**archiving logs**. The process of copying records or logs of IMS activity from the online log data set, which is temporarily recorded on DASD, to the system log data set, which is stored on DASD, tape, or mass storage.

**archive process**. The process of archiving the IMS/ESA OLDS to a system log data set (SLDS) on either tape or DASD.

**area**. A subset of a DEDB that is defined as a VSAM ESDS data set. Each area in a DEDB consists of a root-addressable part, an independent-overflow part, and a sequence-dependent part. Area contains the entire logical structure for a set of root segments and their dependent segment.

**area data set**. A data set that contains a DEDB area. IMS can maintain up to seven copies of this data set.

**Automated Operator**. An application program that can allow a subset of IMS Operator commands and receive status information on the execution of the commands.

**backout**. The process of removing all the database transactions and manages data.

**batch checkpoint/restart.**. The facility that enables batch processing programs to synchronize checkpoints and to be restarted at a user-specified checkpoint.

**batch image copy**. A copy of a database or area that reflects the state the data at a moment in time when no updates were being made. The Database Image Copy utility (DFSUDMP0) produces batch image copies, which IMS utilities use when recovering from failures.

**batch message processing (BMP) program**. An IMS batch processing program that has access to online databases and message queues. BMPs run online, but like programs in a batch environment, they are started with job control language (JCL).

**batch processing**. (1) Type of data processing in which a number of input items are grouped for processing serially with a minimum of operator intervention and no end-user interaction. (2) Serial processing of computer programs. (3) Pertaining to the technique of executing a set of computer programs so that each is completed before the next program of the set is started.

**batch processing program**. An application program that has access to databases and MVS data management facilities but does not have access to the IMS control region or its message queues.

**buffer invalidation**. A technique for preventing the use of invalid data in an IMS Sysplex data sharing environment. The technique involves marking all copies of data in IMS buffers invalid once a sharing IMS subsystem has updated that data.

**checkpoint**. In IMS, the point at which an application program commits that the changes it has made to a database are consistent and complete and releases database segments for use by other programs. You can request checkpoints at appropriate points in a program to provide places from which you can restart that program if it, or the system, fails.

For an IMS system, a point in time from which the system can start again if a failure makes recovery necessary. The checkpoint is performed by IMS itself.

**cold start**. The starting of IMS when it is initialized for the first time or when some error condition prevents a warm or emergency restart.

**command**. A request from a terminal or AO (automated operator) to perform a specific IMS service, such as altering system resource status or displaying specific system information.

**commit**. To make changes permanent for a resource in order to establish a new consist status.

**control region**. The MVS main storage region that contains the IMS control program

**Data Language/I (DL/I)**. The IMS data manipulation language, which is a common high-level interface between a user application and IMS. DL/I calls are invoked from application programs written in languages such as PL/I, COBOL, VS Pascal, C, and Ada. It can also be invoked from assembler language application programs by subroutine calls. IMS lets the user define data structures, relate structures to the application, load structures, and reorganize structures.

**data-sharing**. The concurrent access of databases by two or more IMS subsystems. The IMS subsystems can be in one processor or in separate processors. They can share data at two levels: database level and block level.

**database control (DBCTL)**. An environment allowing full-function database and DEDBs to be accessed from one or more transaction management subsystems.

**database data set (DBDS)**. A data set containing all or part of a database.

**database description (DBD)**. The collection of macro parameter statements that define the characteristics of a database, such as the database′s organization and access method, the segments and fields in a database record, and the relationship between types of segments.

**database recovery**. The process of restoring a physically damaged DBDS by merging an image copy and logs or change accumulations.

**Database Recovery Control (DBRC)**. A feature of the IMS Database Manager that facilitates easier recovery of IMS databases. DBRC maintains information required for database recoveries, generates recovery control statements, verifies recovery input, maintains a separate change log for database data sets, and supports sharing of IMS databases and areas by multiple IMS subsystems.

**DEDB**. A direct-access database that consists of one or more areas, with each area containing both root segments and dependent segments. DEDBs use a data structure that allows them to be used for both hierarchic processing and journaling. The database is accessed by using VSAM′s Media Manager.

**DL/I address space**. An address space used by the online IMS control program to contain most of the DL/I code and control blocks. This option can be selected for the online IMS environment to provide an alternative virtual storage configuration.

**dynamic allocation/deallocation**. A function that removes the requirement to allocate IMS databases, area data sets, and certain system data sets through job control language. A data set is allocated during IMS initialization or when it is first used and is deallocated when it is no longer used (that is, closed or stopped).

**Fast Path**. IMS functions for applications that require good response characteristics and that may have large transaction volumes. Programs have rapid access to main-storage databases (to the field level), and to direct-access data entry databases. Message processing is grouped for load balancing and synchronized for database integrity and recovery.

**IFP**. IMS Fast Path program. A type of program designed to operate with expedited message handling (EMH) in a Fast path region.

**IMS subsystem**. An individual batch or online IMS control program executing in an MVS address space.

**IMS system log**. Logically, a single log made up of on-line data sets (OLDSs) and write-ahead data sets (WADSs).

**Intersystem Communication (ISC)**. An extension of IMS Multiple Systems Coupling that permits the connection of IMS to another IMS subsystem, to CICS/MVS, or to a user-written subsystem, provided both subsystems use ISC.

**IRLM**. An IMS component that provides lock management for use by IMS subsystems that share data in an MVS environment.

**lock management**. The reservation of a segment by a program. Other programs are kept from using the segment until the program using it is done.

**MPR**. An IMS/ESA online message processing region.

**Not-IWAIT time**. Elapsed time minus IWAIT time for an event. Approximate CPU up time if there is no interference.

**online**. Applicable in the IMS DB/DC, DBCTL, and DCCTL environments, unless otherwise indicated.

**online change**. The process of adding, deleting, or replacing various IM resources without stopping the system to redefine them.

**online image copy**. The process of creating an image copy while the DBDS is online. Also, the image copy created by the process.

**OLDS**. The IMS/ESA online log data set that contains the log records of all IMS/ESA activity.

**program specification block (PSB)**. The control block that describes databases and logical message destinations used by an application program. A PSB consists of one or more PCBs.

**recovery control (RECON) data sets.**. Data sets in which Data Base Recovery Control stores information about logging activity and events that might affect the recovery of databases.

**RSR**. IMS Remote Site Recovery, which provides remote recovery for IMS DB full function databases, IMS DB Fast Path databases, IMS TM Message Queues, and the IMS TM telecommunications network.

**system log data set (SLDS)**. The system log data set, which is normally the tape data set that contains the archived OLDS.

**transaction**. A message from a terminal or an application program that causes the application program logic to be executed.

**unit of work**. A distinct unit of processing that can be released by the application program for use by other programs in the system.

**XRF**. A software function designed to minimize the impact of various failure of IMS users.

# List of Abbreviations

| | |
|---|---|
| ***ACB*** | Access control block in VSAM. Application control block in IMS. |
| ***ACF/VTAM*** | Advanced communication function/virtual telecommunications access method. |
| ***AMS*** | Access method services. |
| ***AO*** | Automatic operation. |
| ***APF*** | Authorized program facility. |
| ***APPC*** | Advanced program-to-program communication; also advanced peer-to-peer communication. |
| ***APPLID*** | VTAM application ID. |
| ***ARM*** | Automatic restart manager |
| ***BMP*** | Batch message processing (IMS region). |
| ***BTAM*** | Basic telecommunications access method. |
| ***CA*** | Control area. |
| ***CI*** | Control interval. |
| ***CICS*** | Customer information control system. |
| ***CQS*** | Common queue service. |
| ***DASD*** | Direct access storage device. |
| ***DBCTL*** | Database control. |
| ***DBD*** | Database description. |
| ***DBRC*** | IMS/VS Database recovery control. |
| ***DB2*** | Database 2. |
| ***DC*** | Database communication. |
| ***DEADQ*** | Dead letter queue. |
| ***DEDB*** | Data entry database. |
| ***DFSnnnnz*** | IMS message where nnnn are four numerics and the suffix z indicates the user action; I = information, A = action, W = warning. |
| ***DFSxxxxx*** | IMS module, control block, table, or PROCLIB member. |
| ***DNS*** | Domain name server |
| ***DCB*** | Data set control block. |
| ***DD*** | Data set definition. |
| ***DEDB*** | Data entry database. |
| ***DL/I*** | Data language I. |
| ***DLS*** | DL/I separate address space. |
| ***DLISAS*** | DL/I separate address space. |
| ***EMH*** | Expedited message handler. |
| ***EMHB*** | Expedited message handler buffer (terminal buffer). |
| ***ESA*** | Enterprise systems architecture. |
| ***ETO*** | Extended terminal option. |
| ***FDBR*** | Fast database recovery. |
| ***FIFO*** | First in, first out. |
| ***GSAM*** | Generalized sequential access method. |
| ***HD*** | Hierarchic direct. |
| ***HDAM*** | Hierarchic direct access method. |
| ***HIDAM*** | Hierarchic indexed direct access method. |
| ***I/O PCB*** | Input/output program communication block, also called TPPCB. |
| ***IBM*** | International Business Machines Corporation. |
| ***IFP*** | IMS Fast Path (IMS region). |
| ***IMS*** | Information management system. |
| ***IPL*** | Initial program load. |
| ***IRLM*** | IMS resource lock manager. |
| ***ISC*** | Intersystem communication. |
| ***ITSO*** | International Technical Support Organization. |
| ***JCL*** | Job control language. |
| ***LTERM*** | Logical terminal. |
| ***MPP*** | Message processing program. |
| ***MPR*** | Message processing region (region in which MPPs are run). |
| ***MSC*** | Multiple systems coupling. |
| ***MTO*** | Master terminal operator. |
| ***MVS*** | Multiple virtual system. |
| ***OLDS*** | Online log data set. |
| ***OSAM*** | Overflow sequential access method. |
| ***PARMLIB*** | MVS parameter library data set. |
| ***PROCLIB*** | Procedure library data set; IMS procedures are in IMSESA.PROCLIB and MVS procedures are in SYS1.PROCLIB. |
| ***PSB*** | Program specification block, made up of PCBs, both I/O and databas e. |
| ***RAA*** | Root addressable area. |
| ***RACF*** | Resource access control facility. |

| | |
|---|---|
| ***RECON*** | Recovery control. |
| ***RSR*** | Remote site recovery. |
| ***SLUP*** | Service logical unit parameter. |
| ***SLUTYPEP*** | Service logical type parameter. |
| ***SMP*** | System modification program. |
| ***SNA*** | System network architecture. |
| ***SPA*** | Scratch pad area. |
| ***TCB*** | Task control block. |
| ***TCO*** | Time Controlled Operations scripts. |
| ***TCP/IP*** | Transmission control protcol/Internet protocol. |
| ***TM*** | Transaction manager. |
| ***TSO*** | Time sharing option. |
| ***UOW*** | Unit of work. |
| ***VSAM*** | Virtual sequential access method. |
| ***VSO*** | Virtual storage option. |
| ***VTAM*** | Virtual telecommunications access method. |
| ***WLM*** | Work load manager. |
| ***XCF*** | Extended control facility. |
| ***XRF*** | Extended recovery facility. |

# Index

## Special Characters
/CHE command 57
/DBD command 17, 186
/DBR command 17, 179, 186
/DISPLAY command 187
/MODIFY command 184
/MSASSIGN command 75
/OPNDST command 76
/QUIESCE command 57
/START command 16, 34, 186
/STOP command 16, 17, 186

## Numerics
0001 136
0010 136
0014 136
0018 136
0020 136

## A
abbreviations 249
ACBLIB 38, 184
acronyms 249
APPC 58, 131, 159
  GRAFFIN parameter 131
  ISTEXCGR 131
  Resource Resolution Exit 131
APPC/IMS 24, 58, 72, 131
APPC/MVS 24, 58
Application 14, 41, 50, 81, 88
  Application program 14, 32, 62
  Cloned applications 42
  Partitioned 42
  Workload balancing 53, 81
Application databases 41
  Partitioned 41
ARM 25, 75, 126, 133, 136, 200, 201
  CQS 136
  FDBR 137
  Restart groups 134

## B
bibliography 241
BMP 7, 16, 23, 25, 62, 121, 133, 171, 174, 201
  ALTID parameter 175
  BMP backout 25
  BMP restart 25

## C
CF connection 203, 204
CFIRLM 113
CFNAMES 113
Change accumulation 198
CICS 23, 27, 42, 55, 68, 134, 201
  Transactions routing 42
Command 16
  /DBD 17, 199, 205, 206
  /DBR 17, 198, 199, 205, 206
  /DEQ 61
  /DIS 61
  /MSASSIGN 75
  /OPNDST 73, 76
  /STA 199, 206
  /START 16, 199, 200
  /STOP 16, 17
Concurrent image copy 114, 197
Control blocks 32
Coupling Facility 13, 19, 28, 98, 139, 141
  cache structure 141
  CFLEVEL 139
  Connection persistence 151
  Coupling facility performance 121
  Coupling facility structure 19
  OSAM buffer invalidation 151
  OSAM structure 141, 142
  Shared queues structure 154
  structure name 28
  Structure persistence 151
  Structure placement 139
  Structure size 151
  Structure sizing 141
  VSAM buffer invalidation 151
  VSAM Structures 141
CQS 19, 23, 27, 31, 125, 126, 133
  CQS checkpoint 37
  CQS data sets 28
  CQS exit 31
  CQS failure 24
  CQS structures 160
  CQSINIT0 126
  CQSIPxxx 23
  Data sets 29
  DFSSQxxx 31
  global structure 126
  group name 126
  initialization process 125
  JCL library 34
  local structure 126
  Parameter library 33

D

Data Sharing 21, 116, 118
  Batch data sharing 21
  Data-sharing group 27
  n-way data sharing 50
Database 14
  Dynamic back out 14
  Fast database recovery 14
  Partitioned database 42
Database recovery 198
DB2 27, 134, 201, 207
DBCTL 7, 14, 22, 110, 134, 200, 203
DBFHAGU0 83
DBRC 16, 17, 34, 36, 38, 111, 197, 202
  ARCHJCL 172
  CAJCL 172
  GENJCL 34, 36, 172
  GENJCL.ARCHIVE 36
  GENJCL.CA 198, 203
  GENJCL.RECOV 198, 203
  ICJCL 172
  ICRCVJCL 172
  JCL library 36
  JOBJCL 172
  LOGCLJCL 172
  OLIC 172
  OLICJCL 172
  Performance 16
  RECOVJCL 172
  SHARECTL 111
  SHARELVL 111, 197
Dead letter queue 61
DEDB 13, 16, 25, 49, 112, 144, 189, 197, 198, 202
  Data sharing 49
  DEDB lock 123
  DEDB statement 113
  DEDB UOW locks 123
  DEDB VSO 41, 115, 124
  GENJCL 112
  SDEP 49
  Skeletal JCL 112
  VSO area 141
  VSO structure 144, 204
DFCDCxxx 88
DFSAOE00 180
DFSAOUE0 180
DFSCCMD0 14, 89
DFSCMLR0 94
DFSCMPR0 94, 97
DFSCMTR0 56
DFSCTRN0 89
DFSCTSE0 89
DFSCTTO0 105
DFSINSX0 104
DFSMDA 34
DFSMS 198
DFSNDMX0 103
DFSNPRT0 56, 82, 94
DFSSGNX0 59
DFSSQxxx 23, 127
DFSUARC0 36
DFSUDMP0 utility 197
  CIC option 197
DFSUDMT0 utility 197, 198
DFSUICP0 utility 197
DFSUOCU0 184
DFSURGU0 120
DFSVC000 23
Disaster recovery 205, 206, 207
  Image copy only recovery 205
  Latest archived log recovery 206
  Real-Time electronic log vaulting recovery 207
  Time-Stamp recovery 206
DNS 79, 80
Domain Name Server 79
Dynamic allocation 33
  DFSMDA 34
dynamic terminals 97
  RESLIB 97

E

E-MCS 14
ETO 57, 59, 70, 90, 104

F

Fast database recovery 189
FDBR 14, 19, 22, 25, 27, 133, 137, 189
  ARM support 192
  DBRC authorization 192
  Failures 192
  log monitoring 189
  Recovery 191
  Restarting IMS 192
  XCF status monitoring 190
FIFO 110
FORMAT 38, 184

G

Glossary 245

H

HDAM 197, 198
HDAM database 119
  Sources of information 120
HIDAM 197, 198
  HIDAM index 197
HIDAM indexes 122
HIDAM secondary indexes 122
HISAM 197, 198
  HISAM overflow 197

# I

IFP 133
Image copy 197
IMS 5, 27, 48, 87, 141, 167, 170, 179, 183
  /DBD 17, 199, 205, 206
  /DBR 17, 179, 198, 199, 205, 206
  /DEQ 61
  /DIS 61
  /MSASSIGN 75
  /OPNDST 76
  /QUIESCE 57
  /STA 199, 206
  /START 16, 34, 175, 199
  /STOP 16, 17
  ACB library 30, 37, 38
  APAR PQ21039 function 173, 174
  APPC/IMS 24, 58, 131
  APPLCTN macro 103, 110
  Application affinities 62
  AUTH CMD call 89
  Automated operations 179
  Automated restarts IMS 133
  Batch data sharing 21
  Batch message processing region 16
  BMP 7, 23, 25, 171, 174, 201
  BPE configuration 126
  Buffer invalidation 13
  CHNG call 89, 97, 104
  Control blocks 32
  Conversational transaction 62
  CQS 19, 23
  Data sets 28
  Data sharing 5, 13, 109, 201, 206
  Data sharing for DEDB 13
  Data sharing group 27, 209
  Data sharing structure 8
  Data-sharing performance 118
  DATABASE macro 109, 114
  DBC 167
  DBCTL 7, 22
  DBFHAGU0 83
  DBRC 169
  DBRC JCL library 36
  Dead letter queue 61
  DEDB 25, 189
  DFPPBxxx parameter member 168
  DFSAOE00 exit 180
  DFSAOUE0 exit 180
  DFSCCMD0 exit 14, 89
  DFSCMLR0 exit 94
  DFSCMPR0 exit 94, 97
  DFSCTRN0 exit 89
  DFSCTSE0 exit 89
  DFSCTTO0 exit 105
  DFSDCxxx 88
  DFSINSX0 exit 104
  DFSMDA 34
  DFSMDA macro 112

IMS *(continued)*
  DFSMPR 170
  DFSNDMX0 exit 103
  DFSNPRT0 82
  DFSNPRT0 exit 94
  DFSPBxxx 24
  DFSSQxxx 23, 31, 127
  DFSVC000 173, 175
  DFSVNUCx 30
  DFSVSMxx parameter 113
  DFSWTnnn 170
  directed routing 94
  DLISAS 168
  DXRJPROC 170
  Dynamic allocation 33
  Dynamic ETO 72
  dynamic terminals 97
  dynamic transaction 94
  ETO 53, 57, 70, 90, 104
  Executed procedure 170
  Fast database recovery 14
  Fast Path EMH 50
  FDBR 19
  FORMAT 38
  FPUTIL 170
  GENJCL.ARCHIVE command 183
  global automation 179
  ICMD call 89
  IFP 171
  IMS CF access 123
  IMS clone 65
  IMS connection 159
  IMS data sets 28
  IMS in Parallel Sysplex 13
  IMS network 48
  IMS partition 65, 75
  IMS TM 45, 48, 49, 87, 110
  IMS workload router 81
  IMS/ESA 13, 82
  IMS.PROCLIB 167
  IMSBATCH 170
  IMSCTRL macro 23, 109, 168
  IMSFP 170
  IMSGROUP 174
  IMSMSG 170
  IMSplex 47, 66
  IMSRDR 169
  IMSTFMTx 38
  Interactive Fast Path region 16
  IO-PCB 58
  IRLM 16, 19, 110, 170
  ISRT call 97, 104
  JCL library 34
  LINEGRP macro 37
  local automation 179
  log archive 183
  log management 183
  LTERM 15, 98, 104

IMS *(continued)*
- Manages affinity 70
- Master terminal 87, 88
- MATRIX 39
- Message processing region 16
- Message queues 37
- MODBLKS 39
- MSC 15, 42, 81, 82
- MSC logical link 99
- MSC physical link 99
- MSDB 41
- MSLINK macro 91
- MSNAME macro 90, 91, 94, 97
- MSPLINK macro 90, 98
- n-way data sharing 50
- OLDS 25, 37, 207
- Online change 105
- operations 179
- OSAM 13, 113
- OSAM buffer 141
- OSAM structure 141
- Parameter library 33
- Printer sessions 73
- printers 73
- RACF security 160
- RECON 30, 114, 206, 207
- recovery procedures 182, 197
- remote transaction 93, 98
- RESLIB 31, 36, 38, 177
- RLDS 37
- RSR 207
- secondary master terminal 88
- Security 89, 160
- SECURITY macro 89
- Shared queue 5, 13, 51, 56, 65, 66, 81, 83, 91, 125
- Shared queue group 27, 97, 209
- Sign on exit 59
- SLDS 37, 112, 206
- SMU security 161
- SPA 62
- SPOOL macro 105
- support procedures 182
- system data sets 215
- TRANSACT macro 83, 85, 86, 93, 102, 110
- Transaction manager 6, 7
- Undefined destination 103
- VSAM buffer 141
- WADS 37
- workload balance 13
- Workload balancing 66
- XRF 16

IMS batch 7, 19, 200, 203
IMS monitor 120
IMS online 133, 200
IMS time controlled operation 179
IMS workload router 82
IMS.PROCLIB 34
IMSplex 47, 48, 49, 54, 57, 65, 73, 81, 84, 87, 88, 89, 102, 209
- Inter-IMSplex 90
- Intra-IMSplex 88
- Intra-IMSplex MSC 102, 105

IOPCB 61, 104
IPF 16
IRLM 8, 16, 19, 27, 110, 122, 133, 170, 180, 203, 209
- Block lock 122
- Busy locks 122
- Database record lock 122
- IRLM abend 200
- IRLM id 19
- IRLMNM 19, 20
- lock structure 122, 139, 141, 146, 152
- restarting 136
- SCOPE=GLOBAL 111, 122, 180
- SCOPE=LOCAL 111, 114, 122
- SCOPE=NODISCON 111, 115, 122, 180
- SHARELVL 122

ISC 42, 47, 48, 51, 55, 57, 70, 71
- ISC recovery 57
- parallel sessions 56, 57, 70

ISTEXCGR 76, 77, 80
IXCMIAPU utility 139

## J

Job scheduling procedures 183

## L

LINEGRP 87
Lock manager 110
Lock structure 201
LTERM 56, 87, 88, 89, 104
LU 6.1 55
LU 6.2 58, 72
- ETO 59
- Shared queue 59

## M

MATRIX 39, 184
MCS 14
MODBLKS 39, 184
MPR 16, 133
MSC 15, 42, 47, 48, 51, 55, 56, 73, 75, 81, 82, 83, 90, 91, 96, 97
- DFSCMTR0 56
- DFSNPRT0 56
- MSC network 15, 16
- MSNAME 15
- routing exit 56

MSNAME 94
MSPLINK 98
MVS 7, 27, 78, 80
- /*JOBPARM JCL 177
- APPC/MVS 24

MVS *(continued)*
- ARM 75, 127, 200
- CFNAMES 113, 201
- CFRM policy 131, 203
- Checkpoint data sets 125
- Component failure impact analysis 8
- Configuration 7
- Coupling facility 8, 13, 19, 139
- E-MCS 14
- IEFSSNxx 20, 114
- IMS subsystem 19, 22
- ISTGENERIC structure 131
- JCL library 35
- LNKLST 175
- Log stream 125
- LOGR policy 125
- MCS 14
- MVS logger 19, 38
- Online systems 7
- PPT 125
- ROUTE command 198
- Routing 14
- SCHEDxx member 125
- SETSSI command 114
- SETXCF command 133, 152, 160, 201, 202
- Structure recovery data sets 126
- Subsystem 27
- XCF 114

## N

Naming convention 209
- Application data sets 213
- Database data sets 213
- IMS system data sets 213
- Job names 210
- STC names 210
- Structure names 211
- Subsystem name 209
- VTAM APPLIDs 211
- VTAM generic resource group names 211

NETVIEW 180
network balancing 81
Network connectivity 54
Network workload balancing 53

## O

OLDS 25, 37, 207
Online change procedures 184
Online image copy 197
OSAM 13, 113, 124, 161
- cache structure 139
- OSAM buffers 202
- OSAM caching 13, 14
- OSAM database 202, 204
- OSAM structure 202, 204
- structure size 144, 145

## P

Parallel Sysplex 5, 15, 19, 22, 30, 31, 47, 50, 53, 65, 71, 79, 105, 133
Plan 3
- Coupling Facility planning 139
- Develop the plan 8
- Migration plan 5
- Migration planning 7
- Migration tasks 233
- Panning phase 5

Printer 73, 74
Program Properties Table 125

## R

RACF 14, 89, 160
- ACEE macro 89
- FACILITY class 160
- PERMIT command 160
- RDEFINE command 160
- SETROPTS command 160

RECON 30, 37, 38, 111, 114, 199, 206, 207
- RECOVCTL 114
- SHARECTL 114
- SHARELVL 114

Recovery procedures 197
RMF 121
RSR 207

## S

SETXCF command 133
SFM 203
Shared queues 13, 22, 31, 38, 47, 51, 56, 59, 65, 81, 83, 88, 90, 148
- EMH queue structures 149
- EMHQ structure 38
- Initial size 148, 149
- list structure 148
- Maximum size 149
- Message queue structure 148
- Structure maximum size 149
- structure recovery data sets 38

SHISAM 197, 198
SLDS 37, 206
SLUTYPEP 54, 57, 63, 70
SMS 16
- Concurrent copy 16

SNA LU 0 54
SPA 16, 62
- Conversational SPA 16

SYS1.PROCLIB 34
Sysplex Failure Management 203
- REBUILDPERCENT 203
- SFM policy 203

## T

TCP/IP 62, 79
  IP address 79
TERMINAL 87
Time-Stamp recovery 199, 206
  recovery point 199
TN3270 79, 80

## U

U0020 134
U0028 134
U0474 16
U0604 134
U0758 74, 134
U0759 134
U2476 134
U3303 14, 102, 103
U3767 118
User exits 162
  Common queue server exits 164
  IMS system exits 162
  IMS TM exits 163

## V

VSAM 8, 112, 123, 161
  cache structure 139
  CBUF processing 112
  share option 112
  VSAM buffers 202
  VSAM database 202, 204
  VSAM structure 113, 202, 204
VTAM 24, 54, 76, 80, 87
  ACB 64
  ISTEXCUV exit 219
  USERVAR exit 219
VTAM generic resource 7, 13, 22, 24, 53, 55, 63, 66, 69, 71, 74, 76, 80, 131
  affinity management 54
  ETO 53
  Execution option 70
  Generic resource exit 76
  generic resource group 13, 27
  GRSNAME 24
  ISTEXCGR 76, 77, 80
  ISTGENERIC structure 131
  LOGAPPL keyword 74
  resource group 69, 77
  resource group name 131
  Resource resolution exit 69, 70, 74
  resource structure 131

## W

WADS 37
Workload 13
  application workload 65

Workload *(continued)*
  network workload 65
  Workload balance 13, 66, 78, 81

## X

XCF 25, 139, 189, 209
XES 127
XRF 16, 57, 59, 68, 181

## Y

Year 2000 17

# ITSO Redbook Evaluation

IMS/ESA V6 Parallel Sysplex Migration Planning Guide
SG24-5461-00

Your feedback is very important to help us maintain the quality of ITSO redbooks. **Please complete this questionnaire and return it using one of the following methods:**

- Use the online evaluation form found at http://www.redbooks.ibm.com/
- Fax this form to: USA International Access Code + 1 914 432 8264
- Send your comments in an Internet note to redbook@us.ibm.com

Which of the following best describes you?
**__Customer  __Business Partner  __Solution Developer  __IBM employee**
**__None of the above**

**Please rate your overall satisfaction** with this book using the scale:
**(1 = very good, 2 = good, 3 = average, 4 = poor, 5 = very poor)**

**Overall Satisfaction** ____________

Please answer the following questions:

Was this redbook published in time for your needs? Yes____ No____

If no, please explain:
________________________________________________________________________________________________

________________________________________________________________________________________________

________________________________________________________________________________________________

________________________________________________________________________________________________

What other redbooks would you like to see published?
________________________________________________________________________________________________

________________________________________________________________________________________________

________________________________________________________________________________________________

**Comments/Suggestions: (THANK YOU FOR YOUR FEEDBACK!)**
________________________________________________________________________________________________

________________________________________________________________________________________________

________________________________________________________________________________________________

________________________________________________________________________________________________

________________________________________________________________________________________________

SG24-5461-00
Printed in the U.S.A.

IMS/ESA V6 Parallel Sysplex Migration Planning Guide
for IMS TM and DBCTL

SG24-5461-00